https://speakerdeck.com/rmoff/

Streaming ETL in Practice with Oracle, Apache Kafka, and KSQL

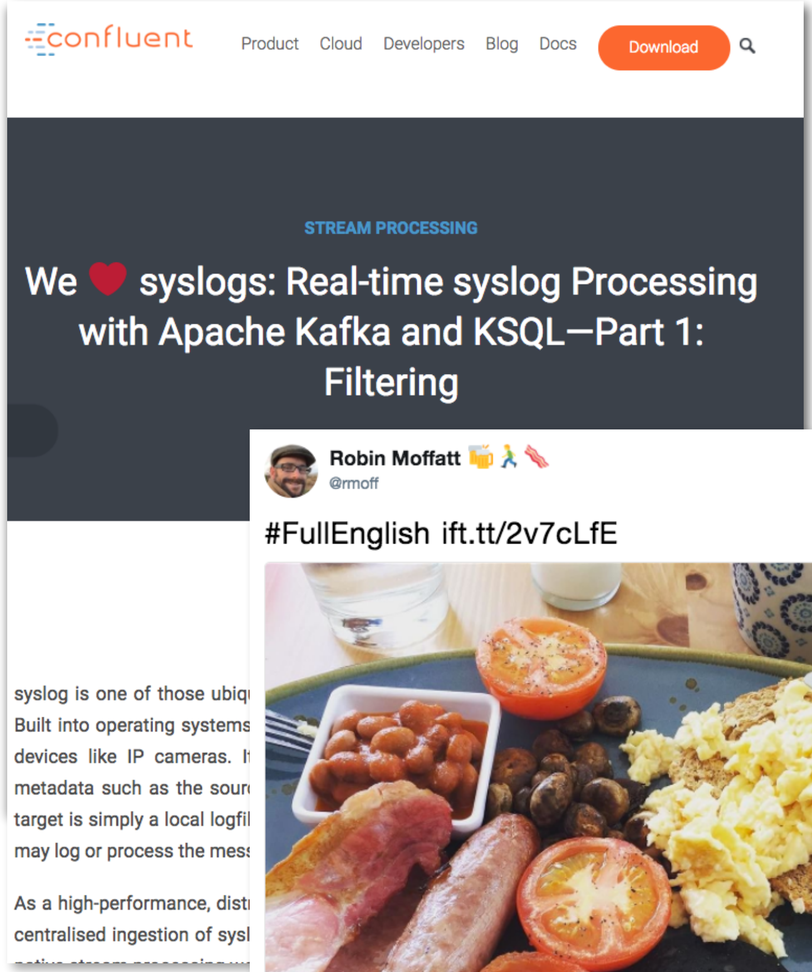
Oracle Developer Meetup

25 Jun 2018 / Robin Moffatt

@rmoff robin@confluent.io

$ whoami

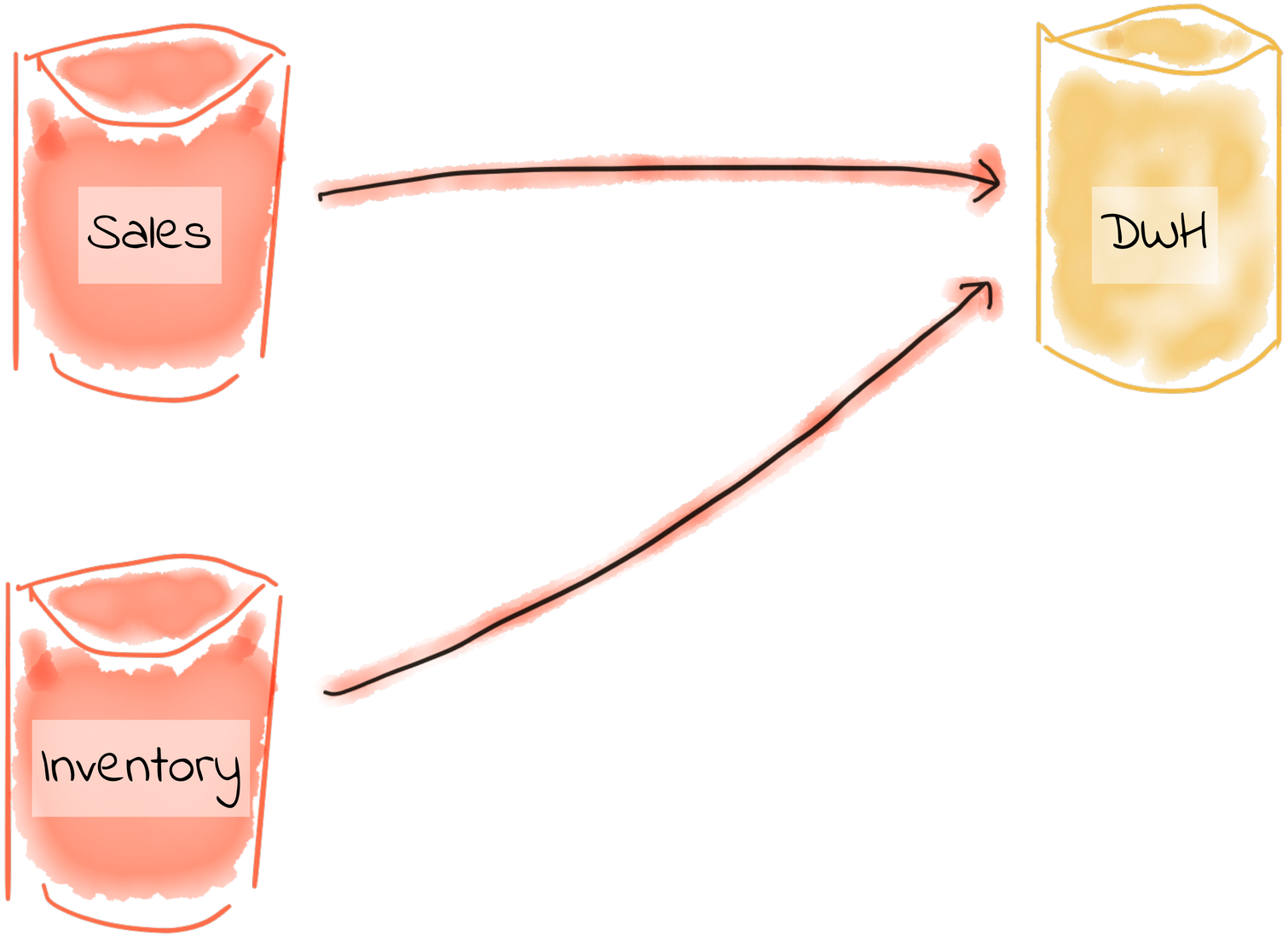
- Developer Advocate @ Confluent
- Working in data & analytics since 2001
- Oracle ACE Director & Dev Champion
- Blogging : http://rmoff.net & http://cnfl.io/rmoff
- Twitter: @rmoff
  - Geek stuff
  - Beer & Fried Breakfasts



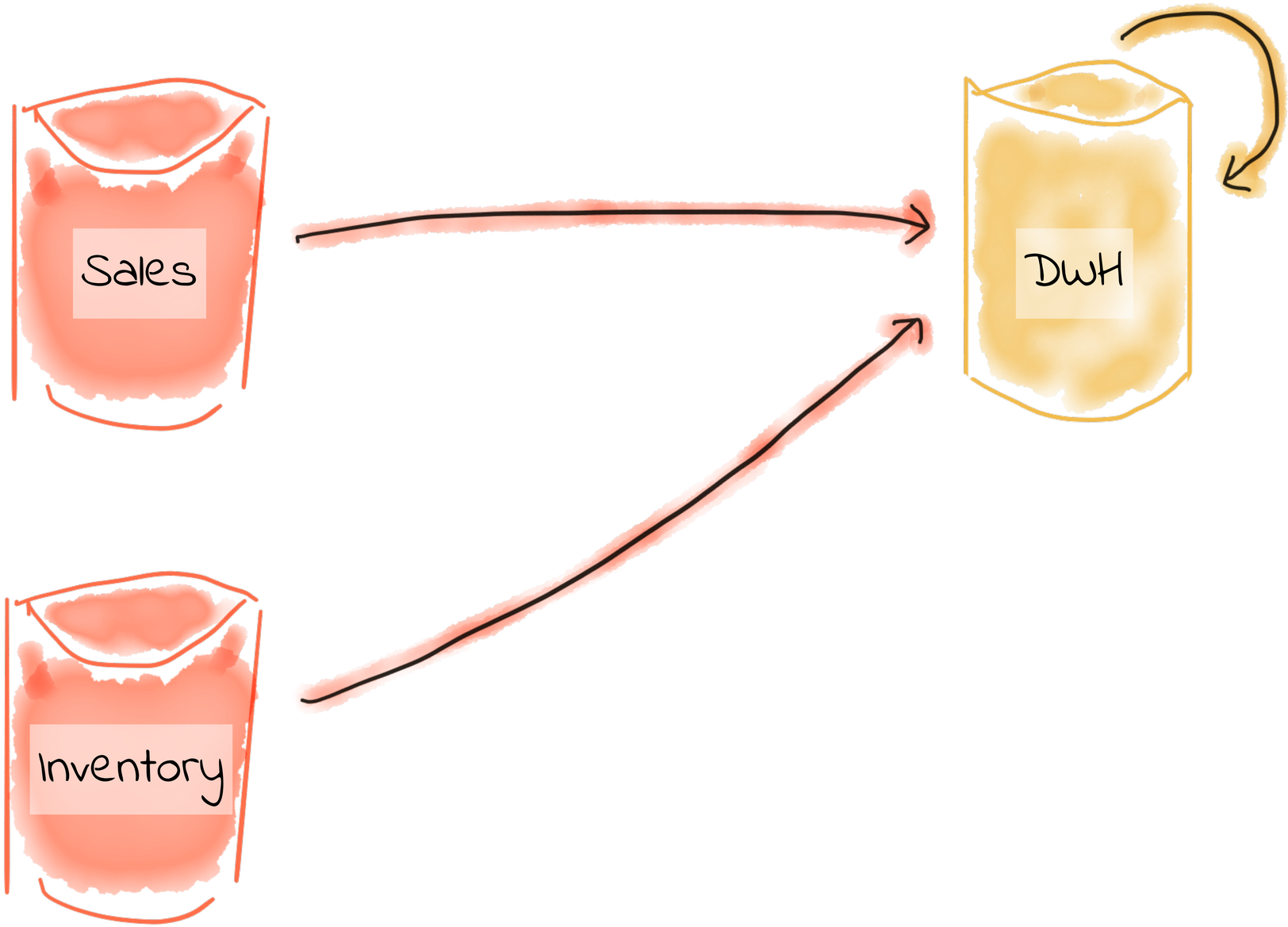
https://speakerdeck.com/rmoff/

Analytics—In the beginning…

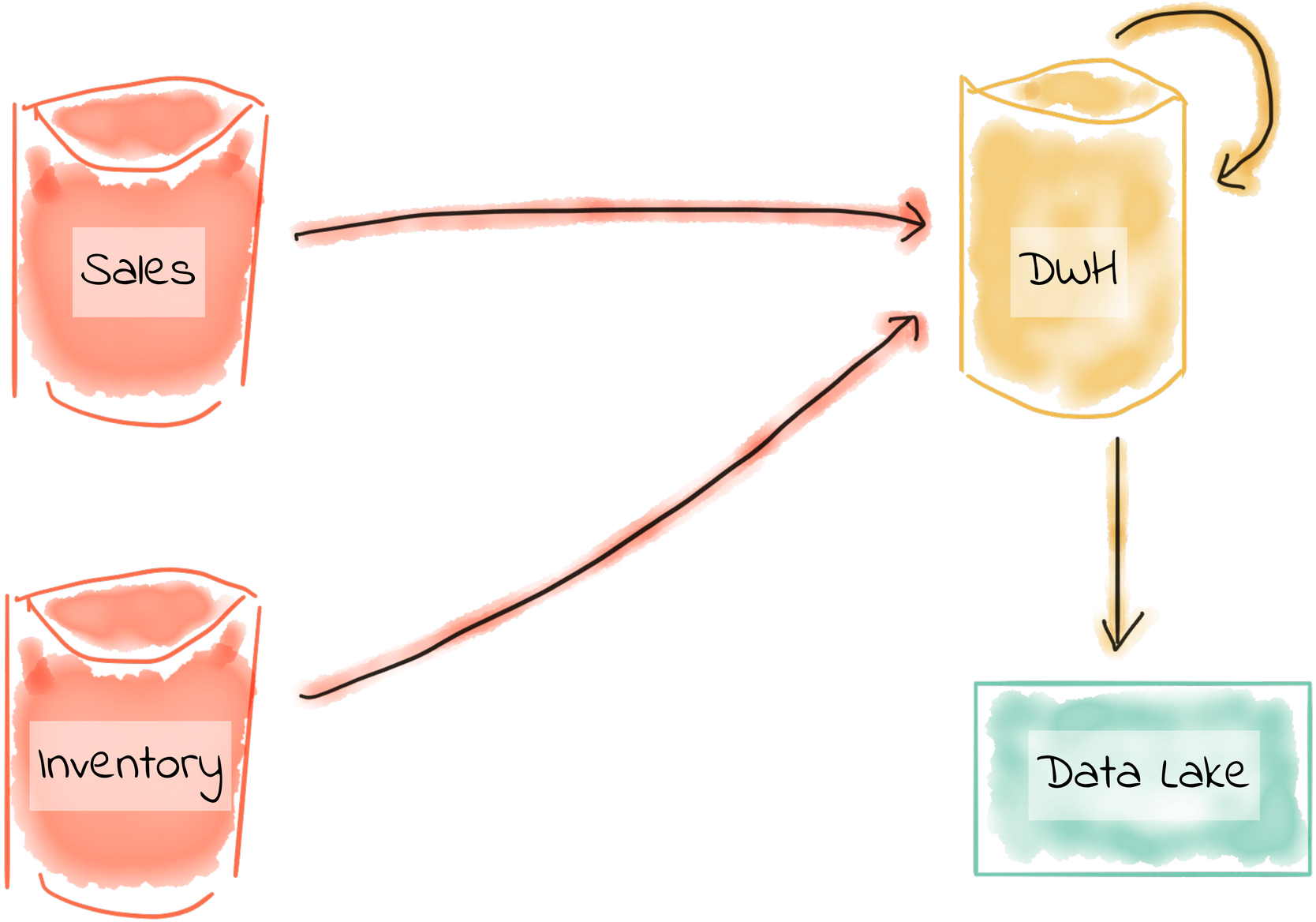
And then there were more data sources…



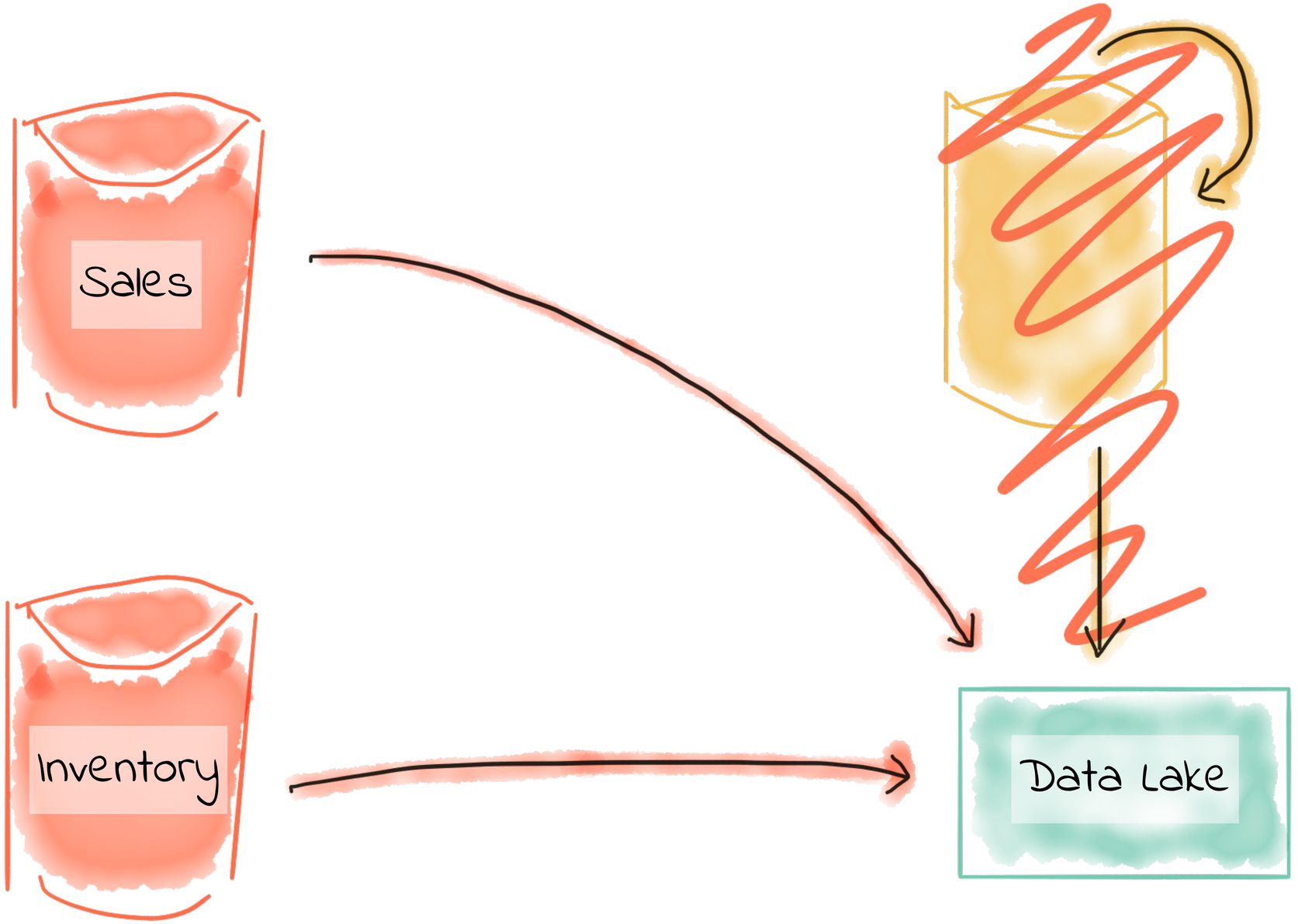
Batch Transformations … (ETL / ELT)



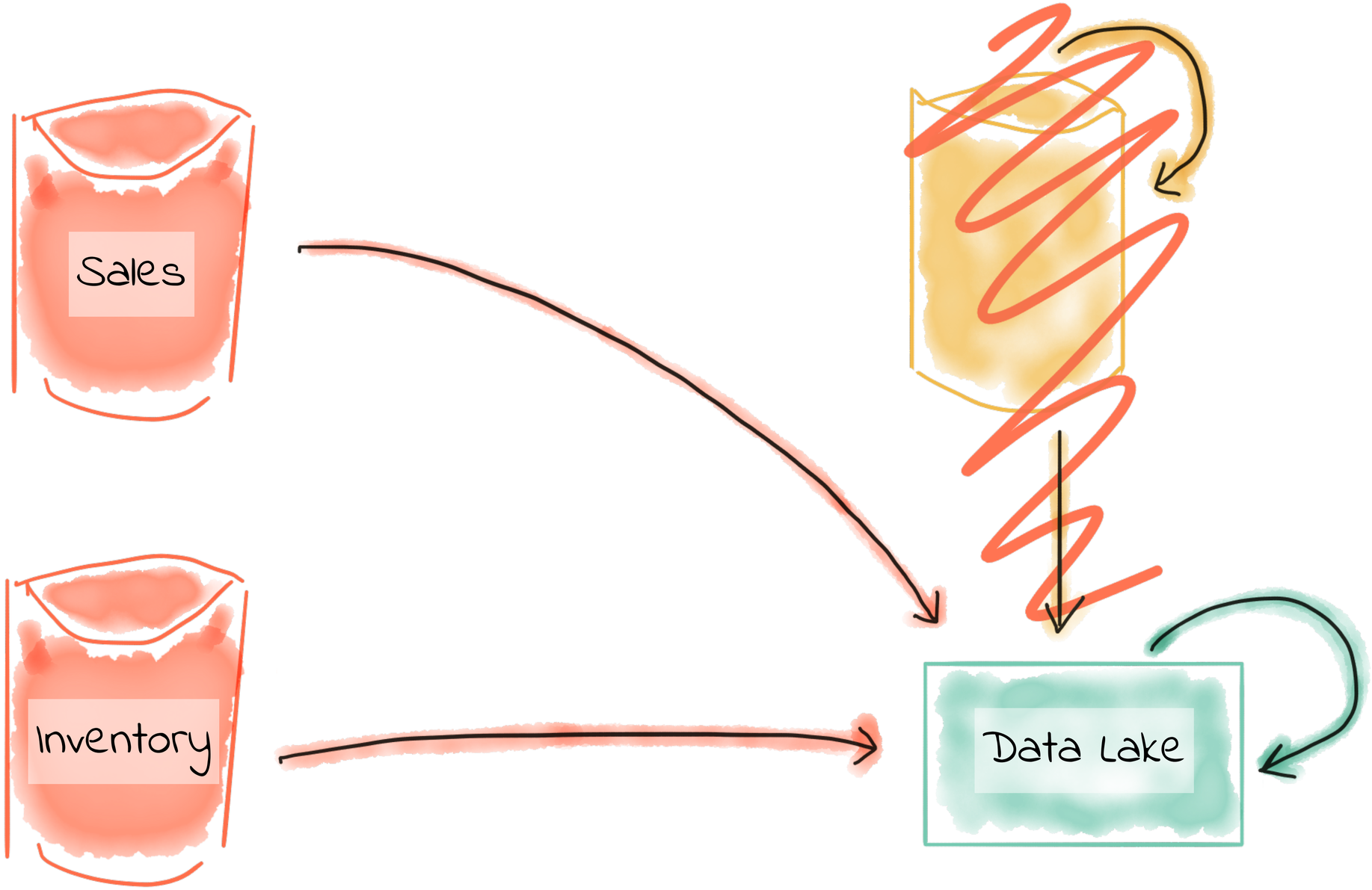
Add a Data Lake…

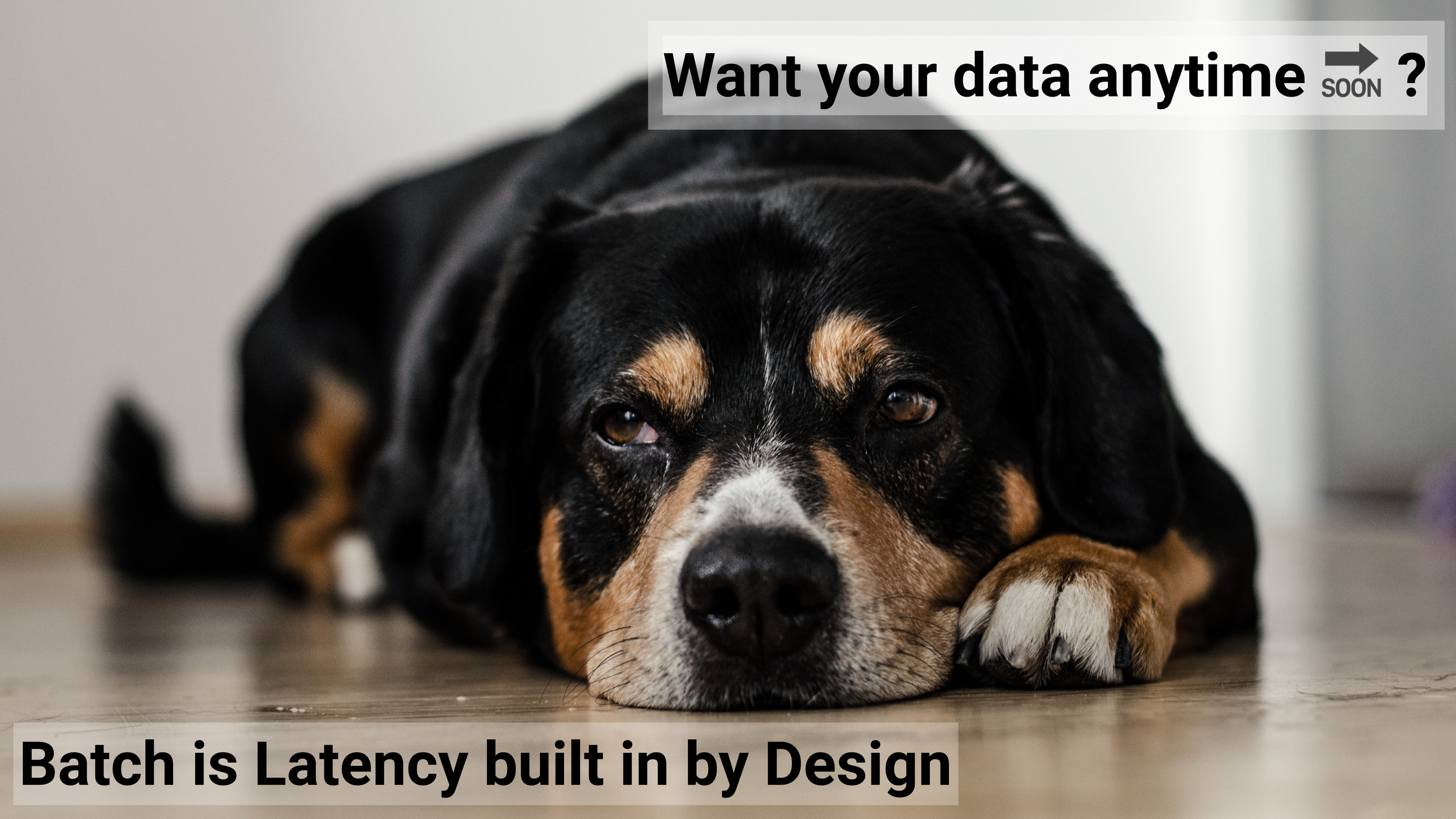


...or Replace the Data Warehouse



Still need to do Batch transformations…




Want your data anytime SOON ?
Batch is Latency built in by Design

The World has Changed

Internet of Things

Microservices

Mobile

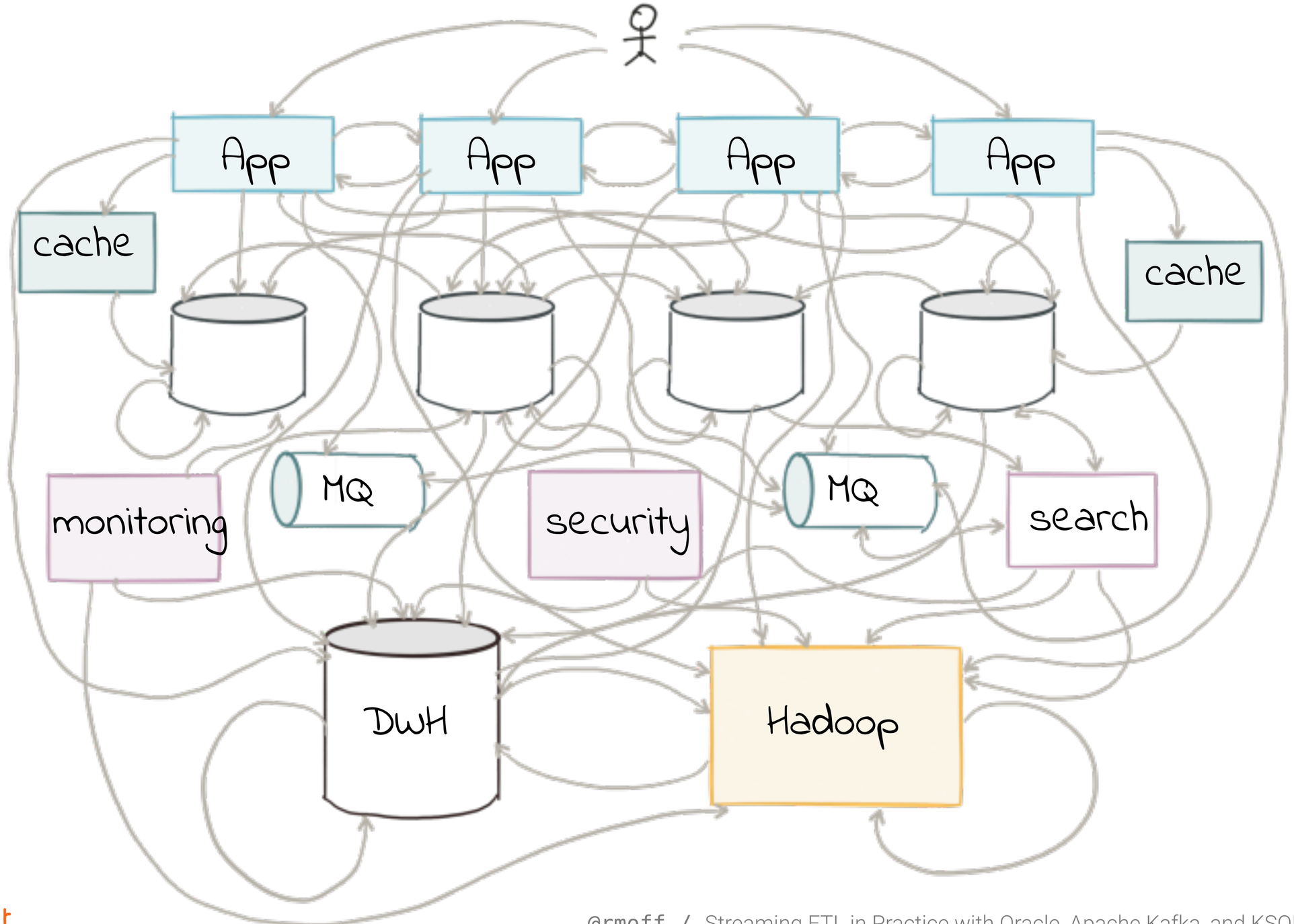
Machine Learning

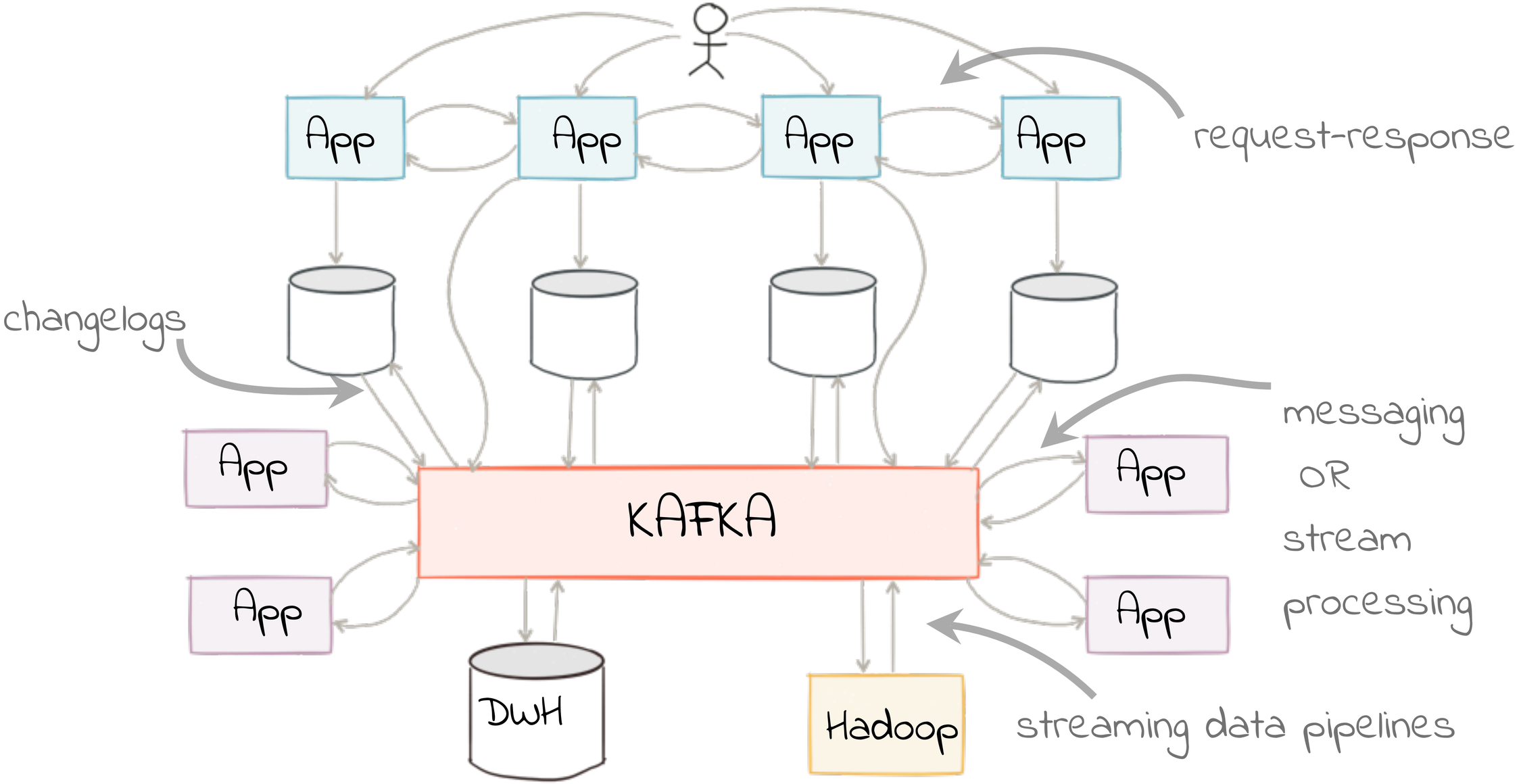
Photo by Denys Nevozhai on Unsplash

Lots of new technologies

(whether you like it or not)

Photo by Rosie Fraser on Unsplash


App
App
App
App
cache
cache
MQ
MQ
monitoring
security
search
DWH
Hadoop


App
App
App
App
request-response
changelogs
App
KAFKA
App
messaging
OR
stream
processing
App
App
DWH
Hadoop
streaming data pipelines

"

But streaming...I've just got data in a database...right?

"Bold claim: all your data is event streams

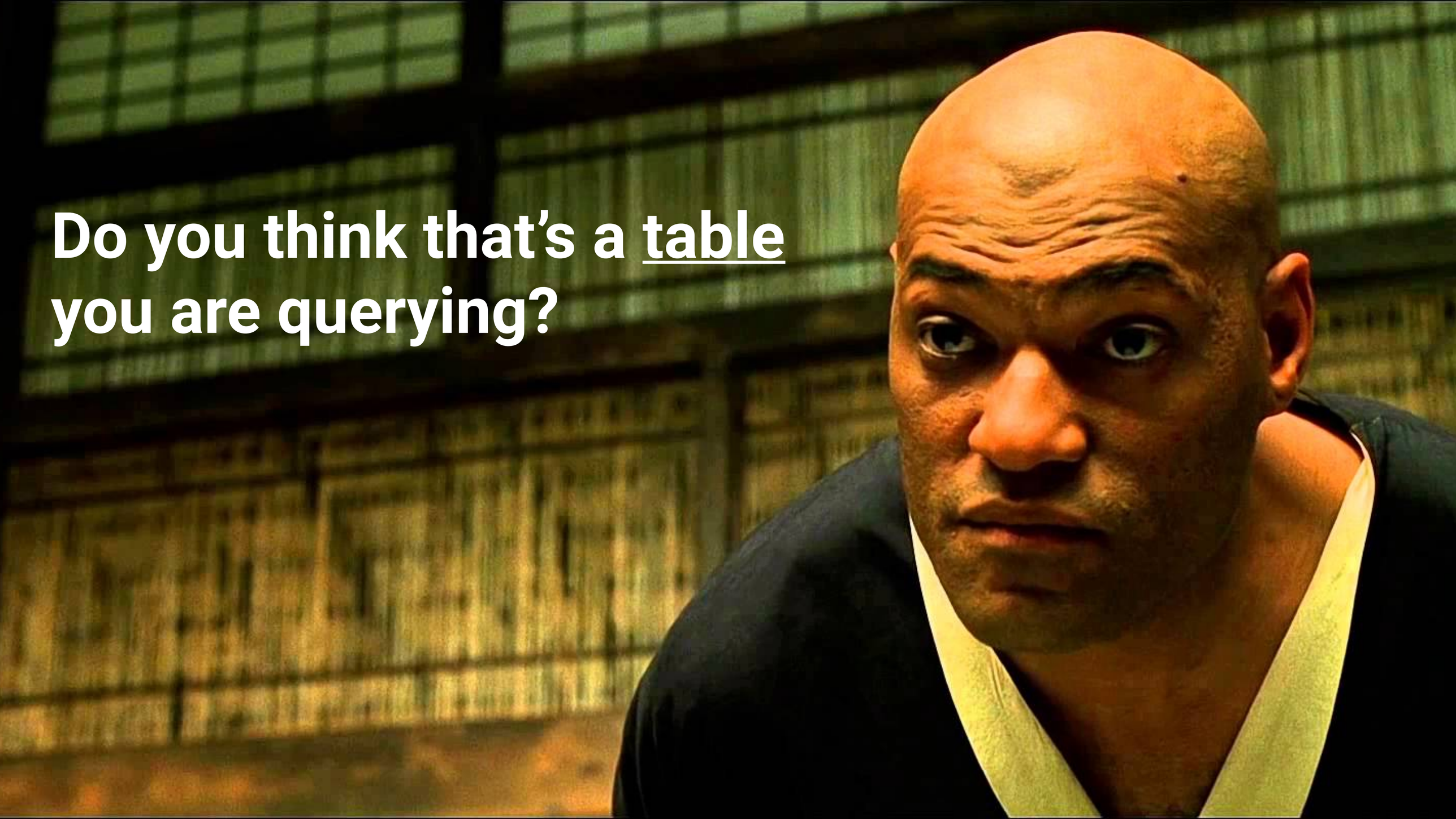
A Customer Experience

A Sale

A Sensor Reading

An Application Log Entry

Databases


Do you think that’s a table
you are querying?

The Table Stream Duality

Account ID	Balance
12345	€50

The Table Stream Duality

Account ID	Amount
12345	+ €50

Time

Account ID	Balance
12345	€50

The Table Stream Duality

Account ID	Amount
12345	+ €50
12345	+ €25

Time

Account ID	Balance
12345	€50

Account ID	Balance
12345	€75

The Table Stream Duality

Time

Account ID	Amount
12345	+ €50
12345	+ €25
12345	-€60

Account ID	Balance
12345	€50

Account ID	Balance
12345	€75

Account ID	Balance
12345	€15

The Table Stream Duality

Stream

Time

Account ID	Amount
12345	+ €50
12345	+ €25
12345	-€60

Table

Account ID	Balance
12345	€50

Account ID	Balance
12345	€75

Account ID	Balance
12345	€15

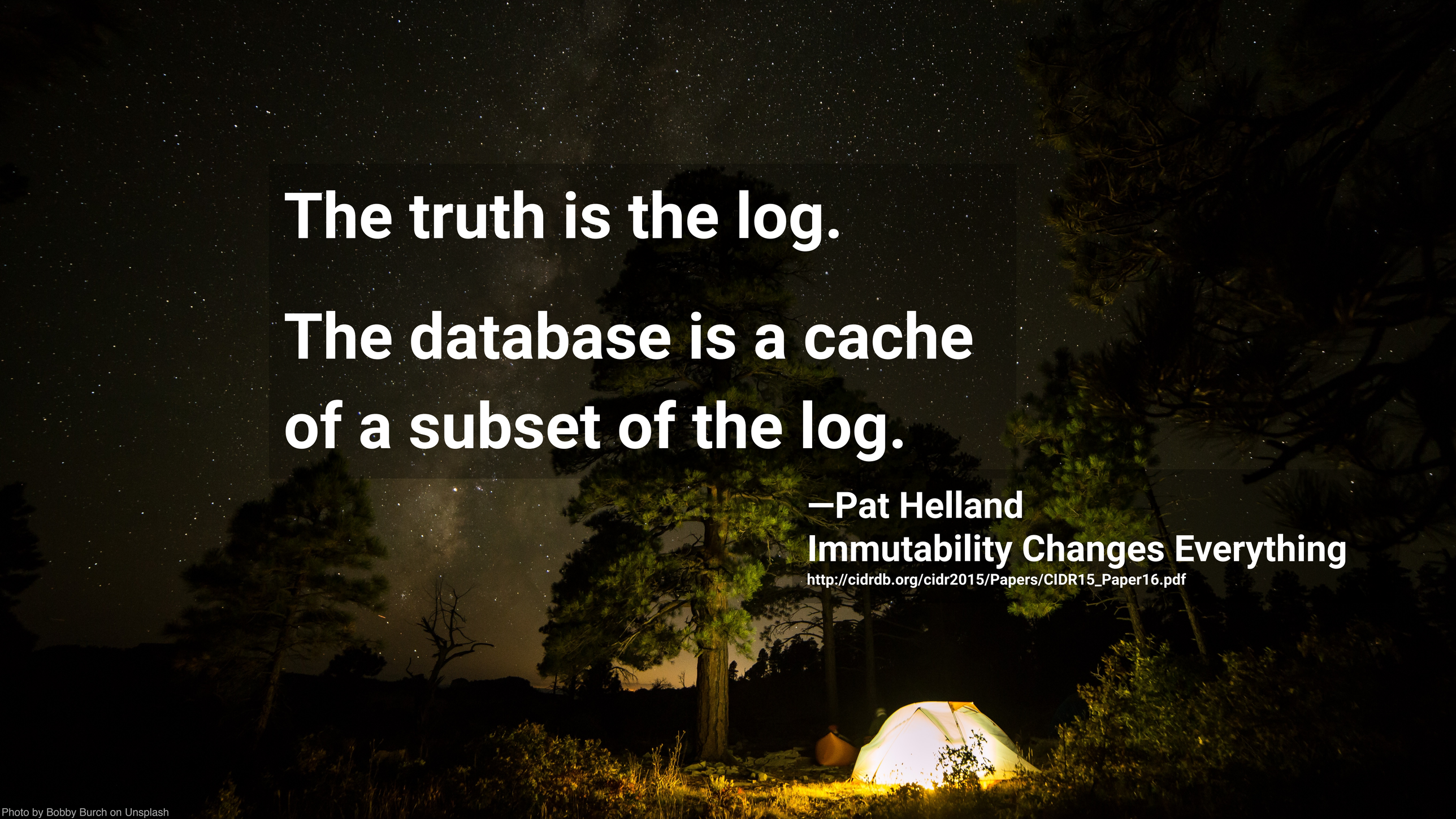
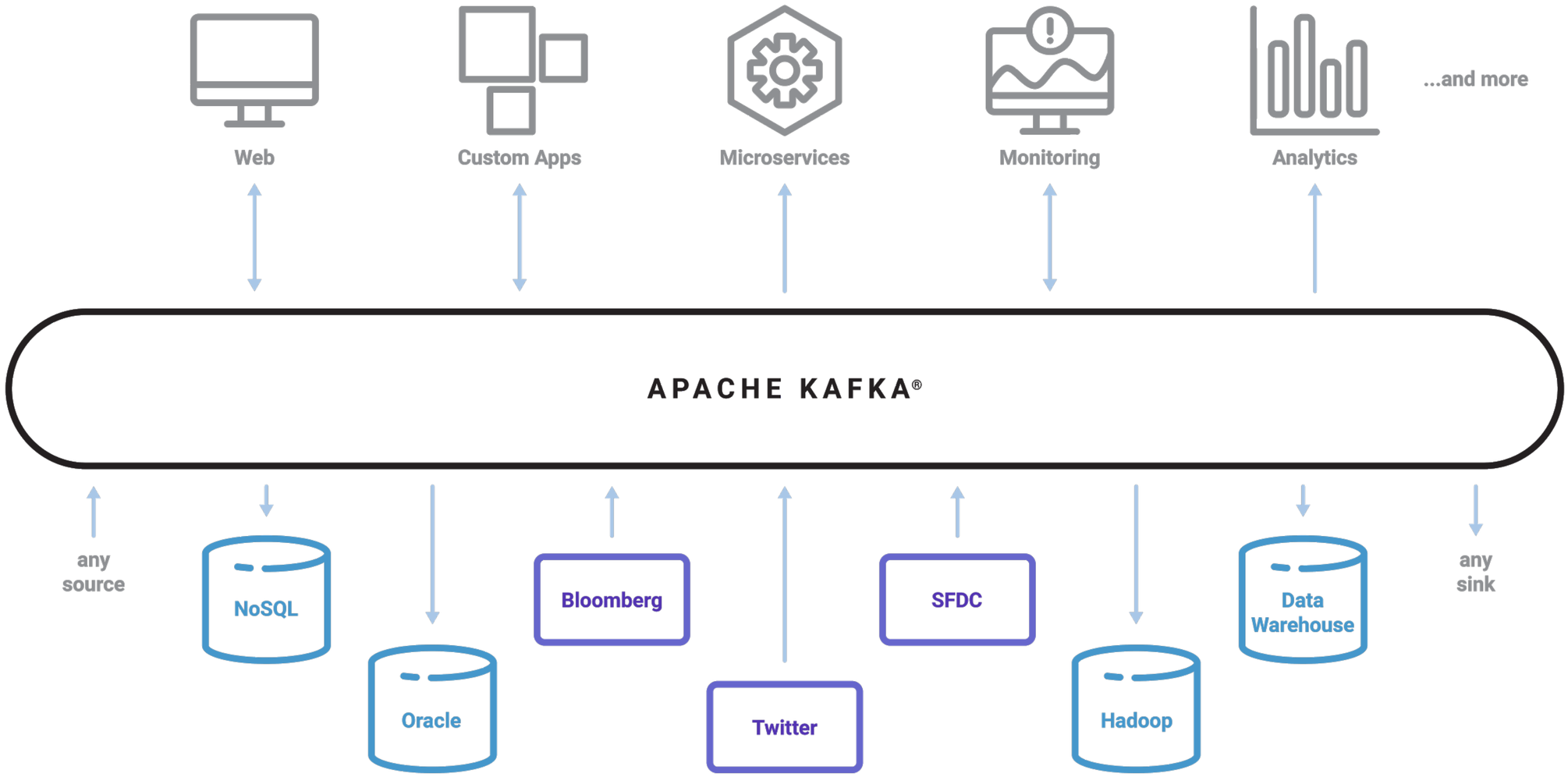


Photo by Bobby Burch on Unsplash

Streaming Platform Vision



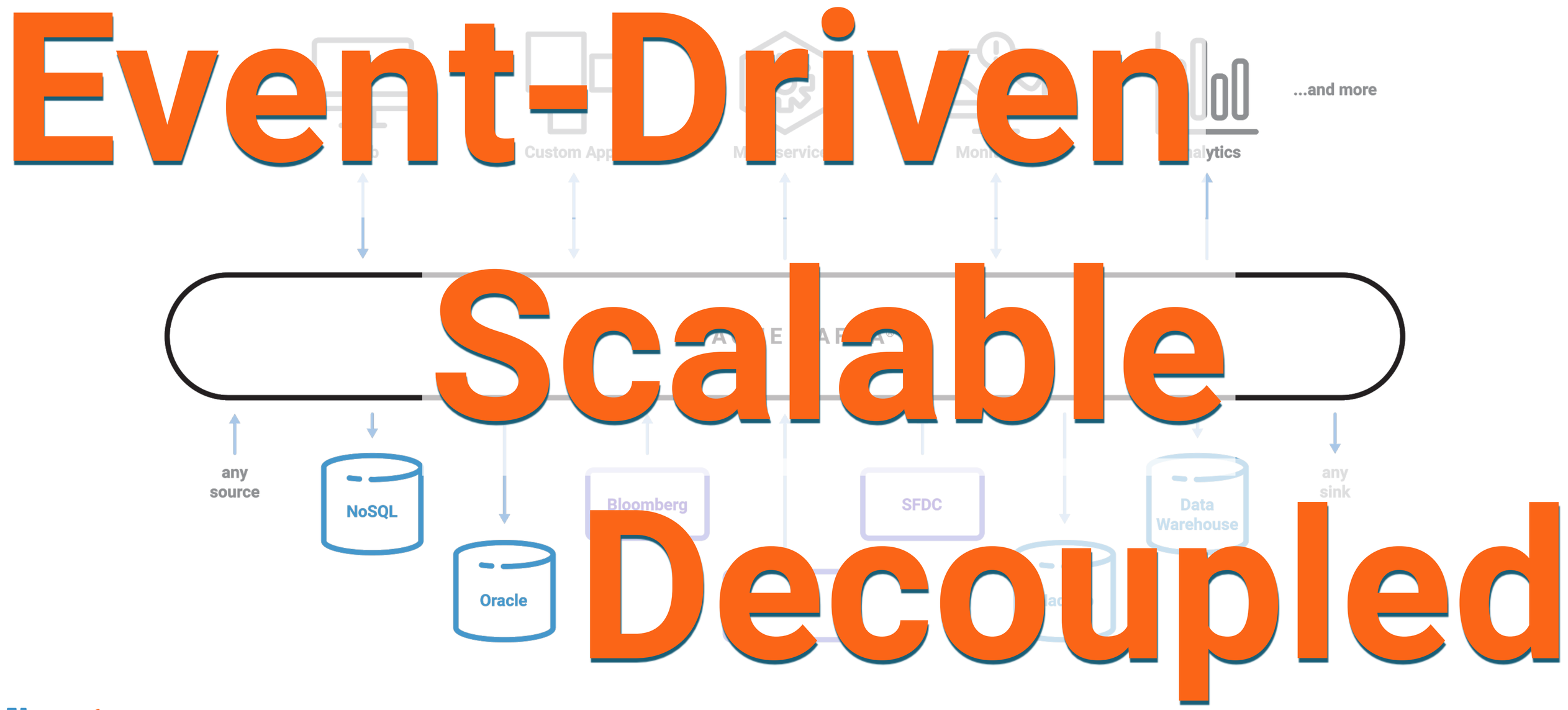
Streaming Platform Vision

Event-Driven

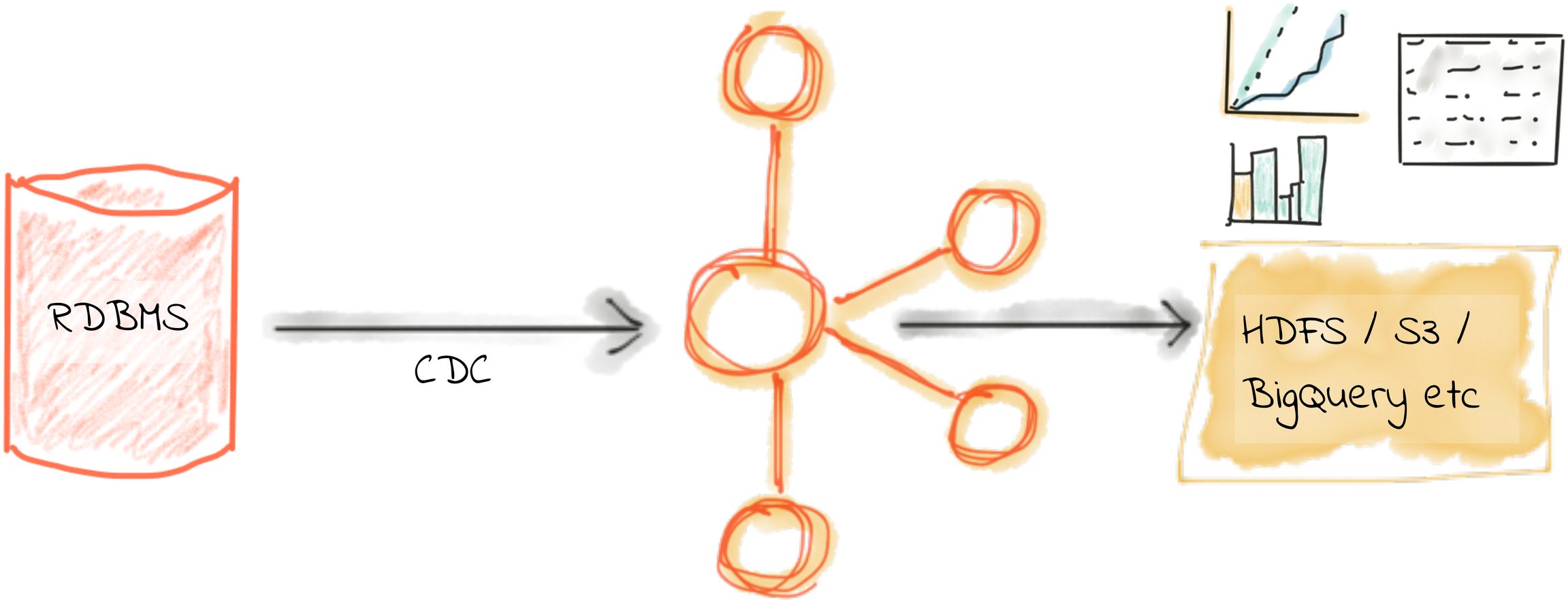
Scalable

Decoupled

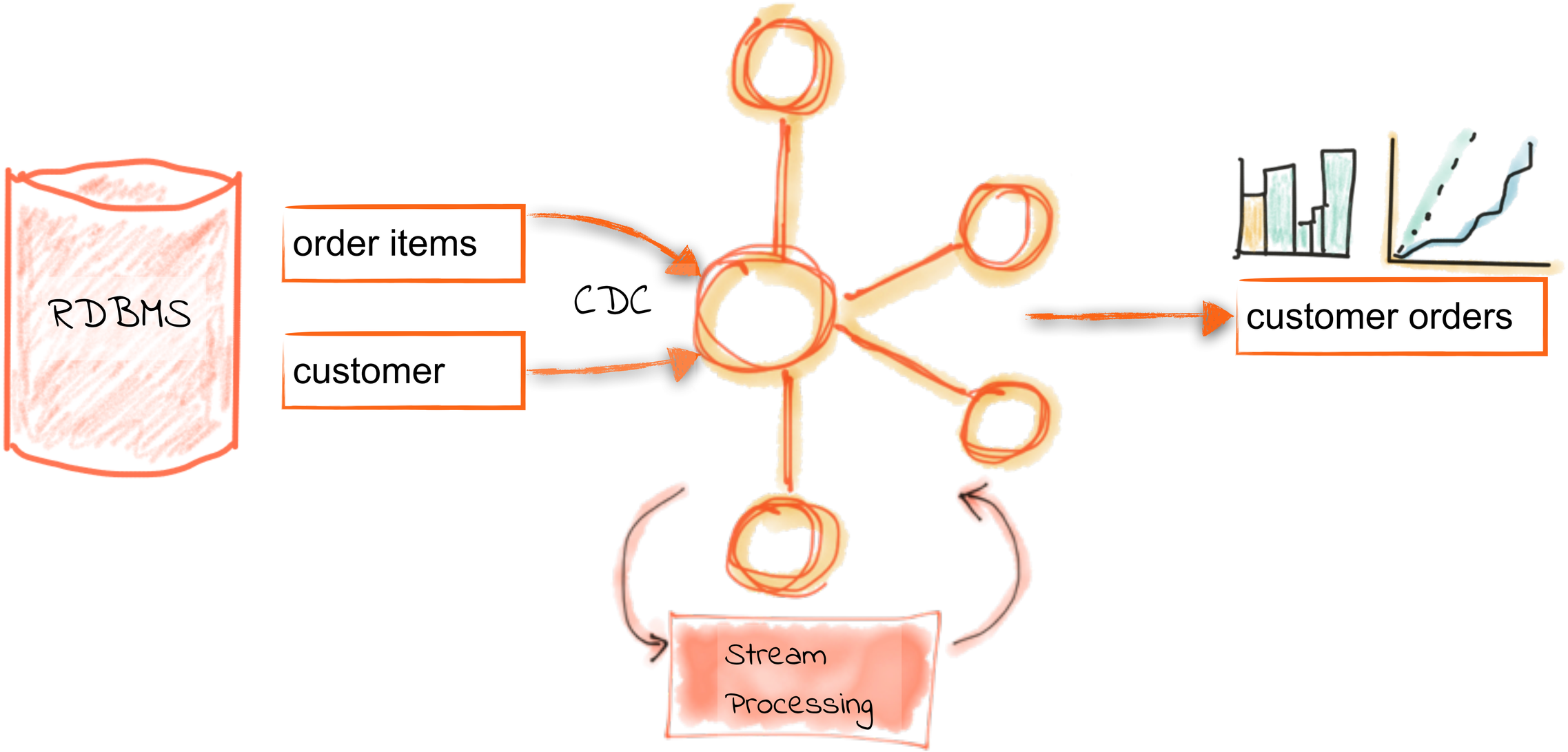
Streaming Platform Vision



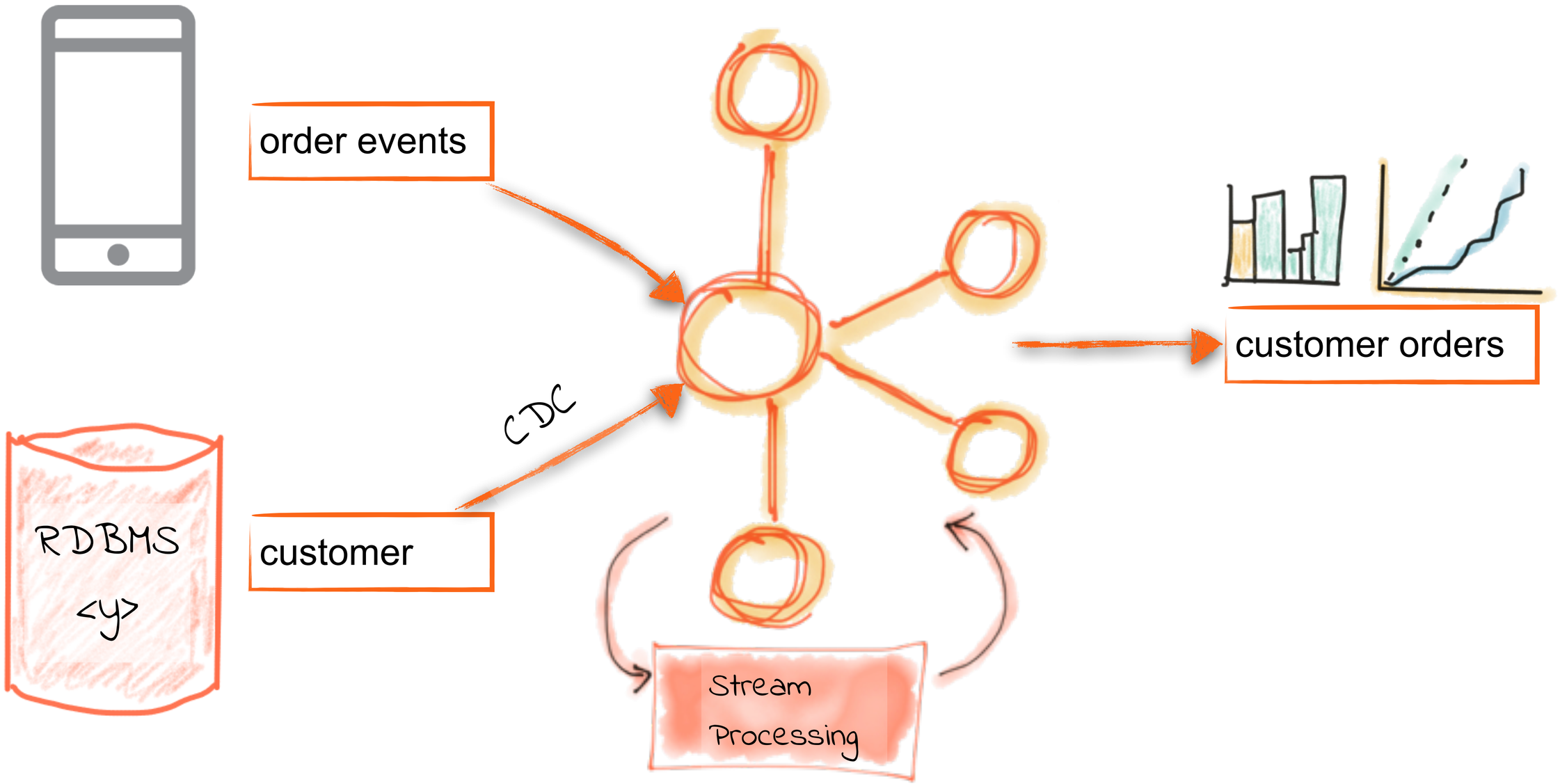
Database offload→Hadoop/Object Storage/Cloud DW for Analytics



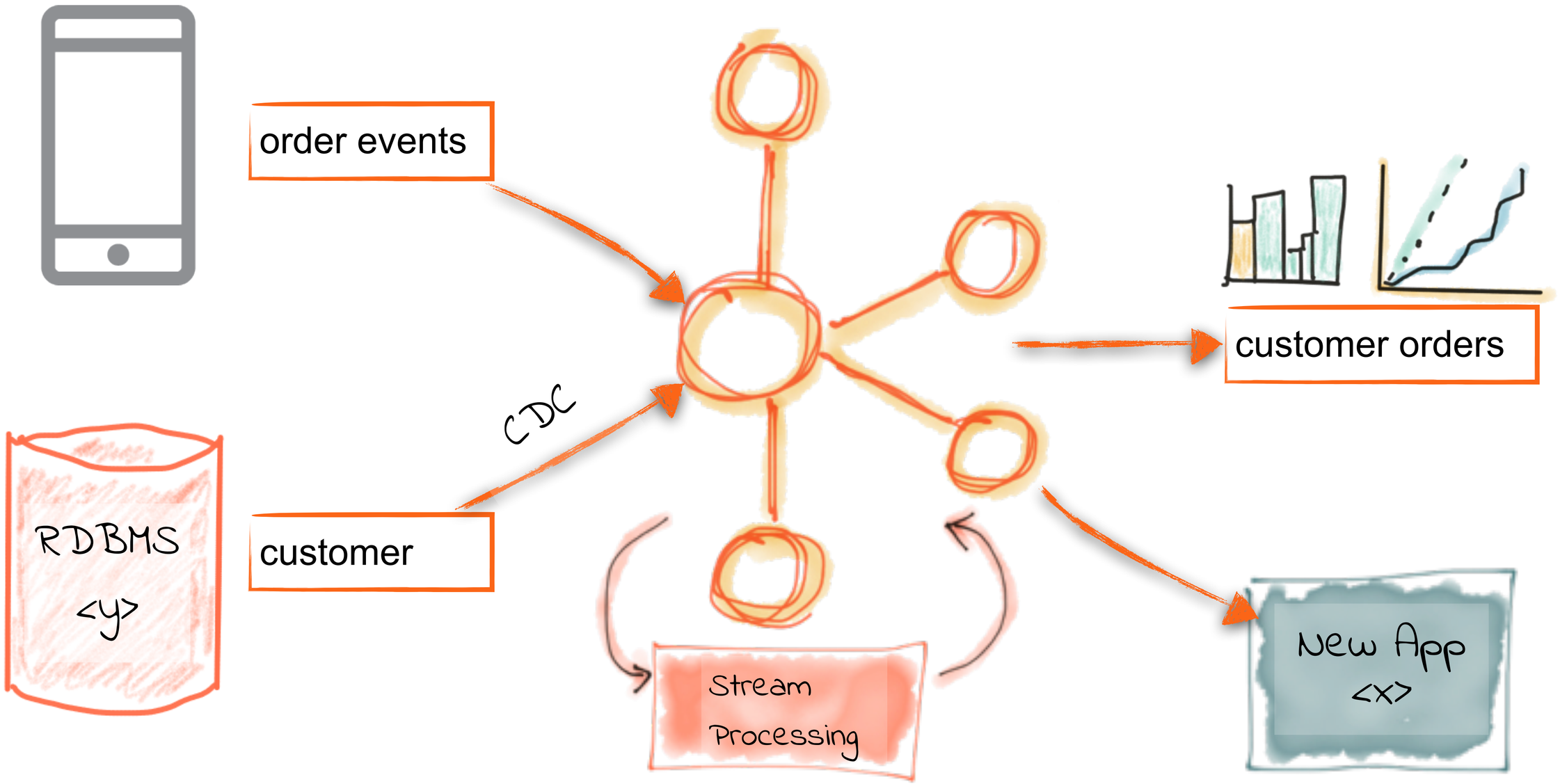
Streaming ETL with Apache Kafka and KSQL



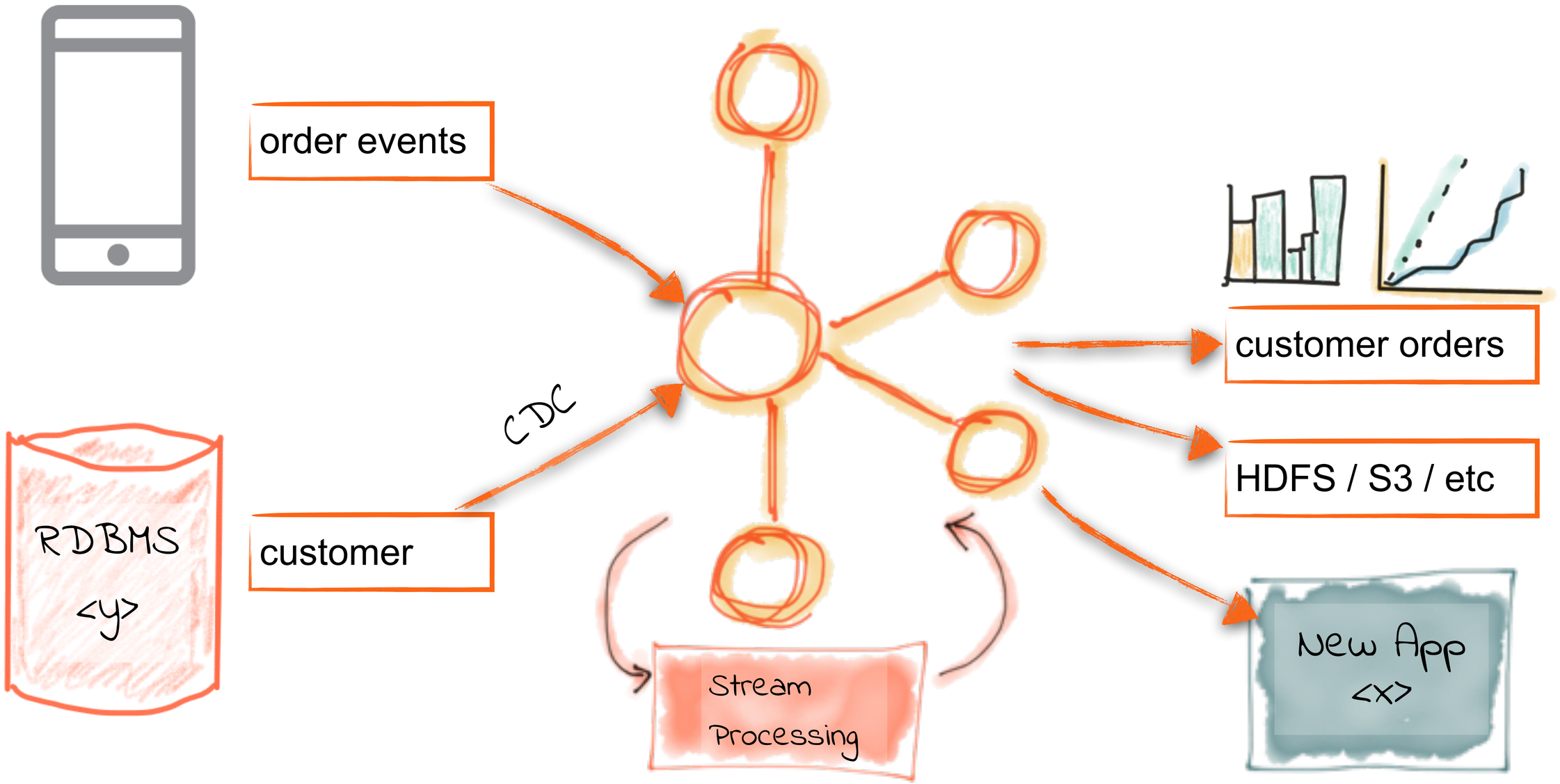
Real-time Event Stream Enrichment with Apache Kafka and KSQL



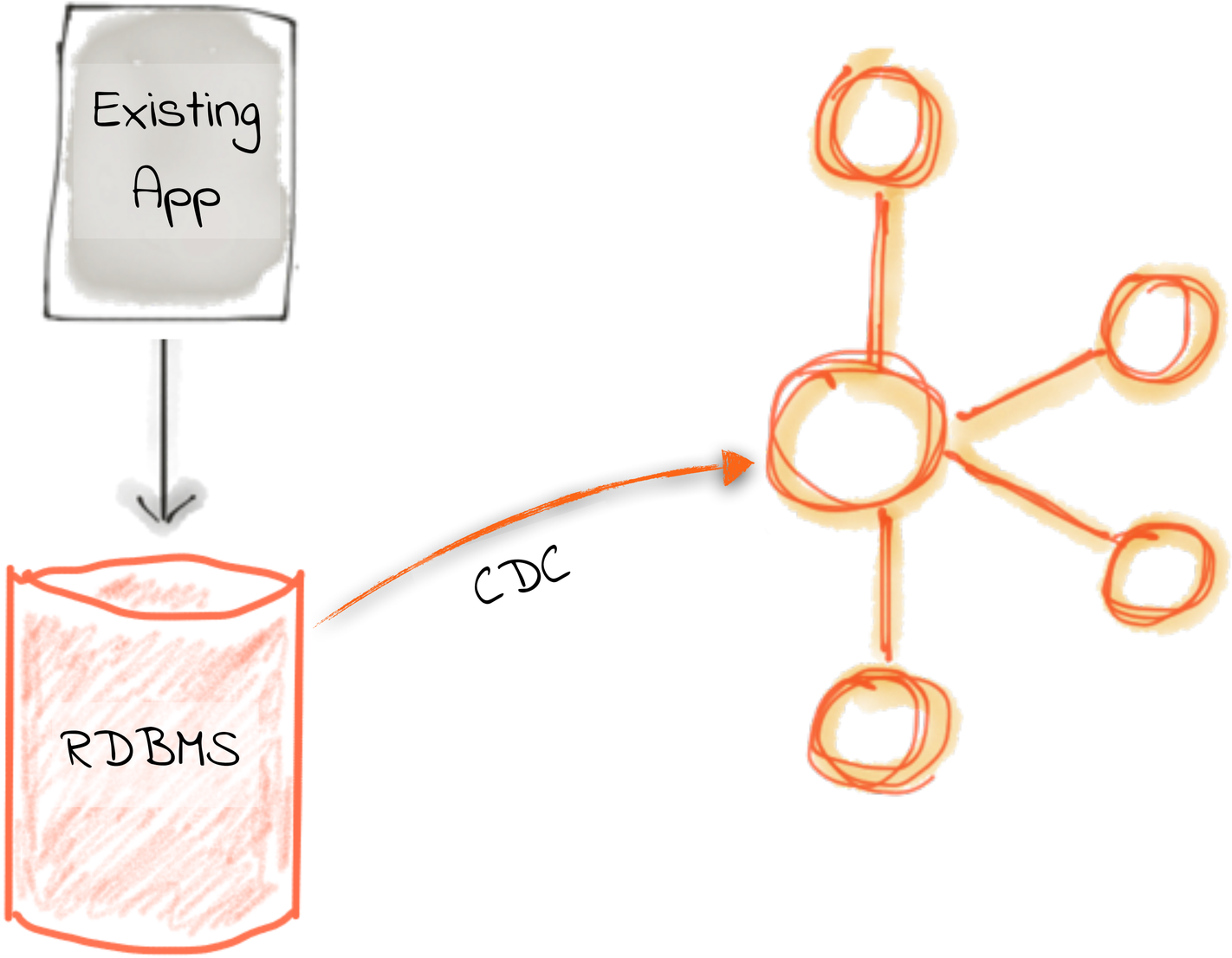
Transform Once, Use Many



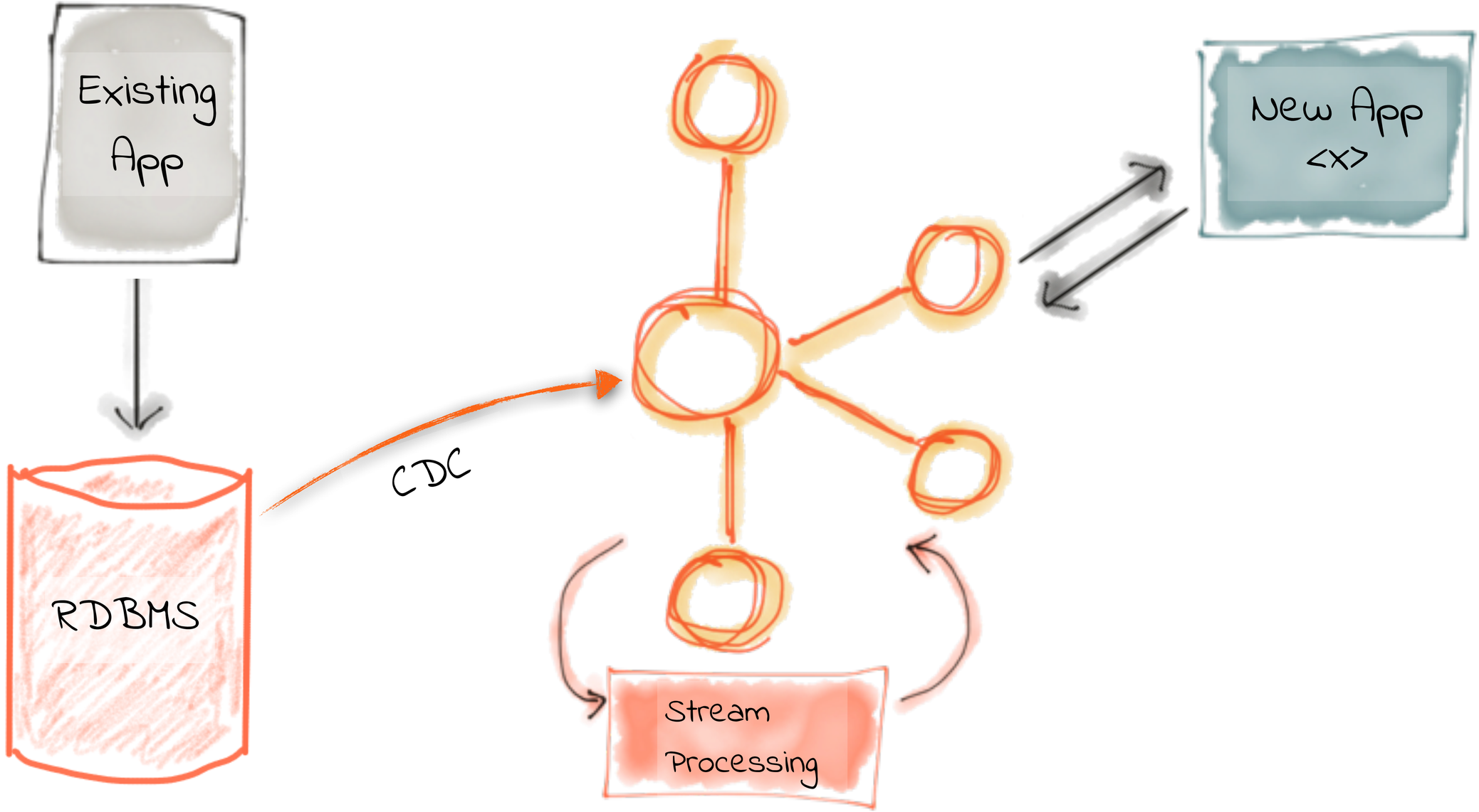
Transform Once, Use Many



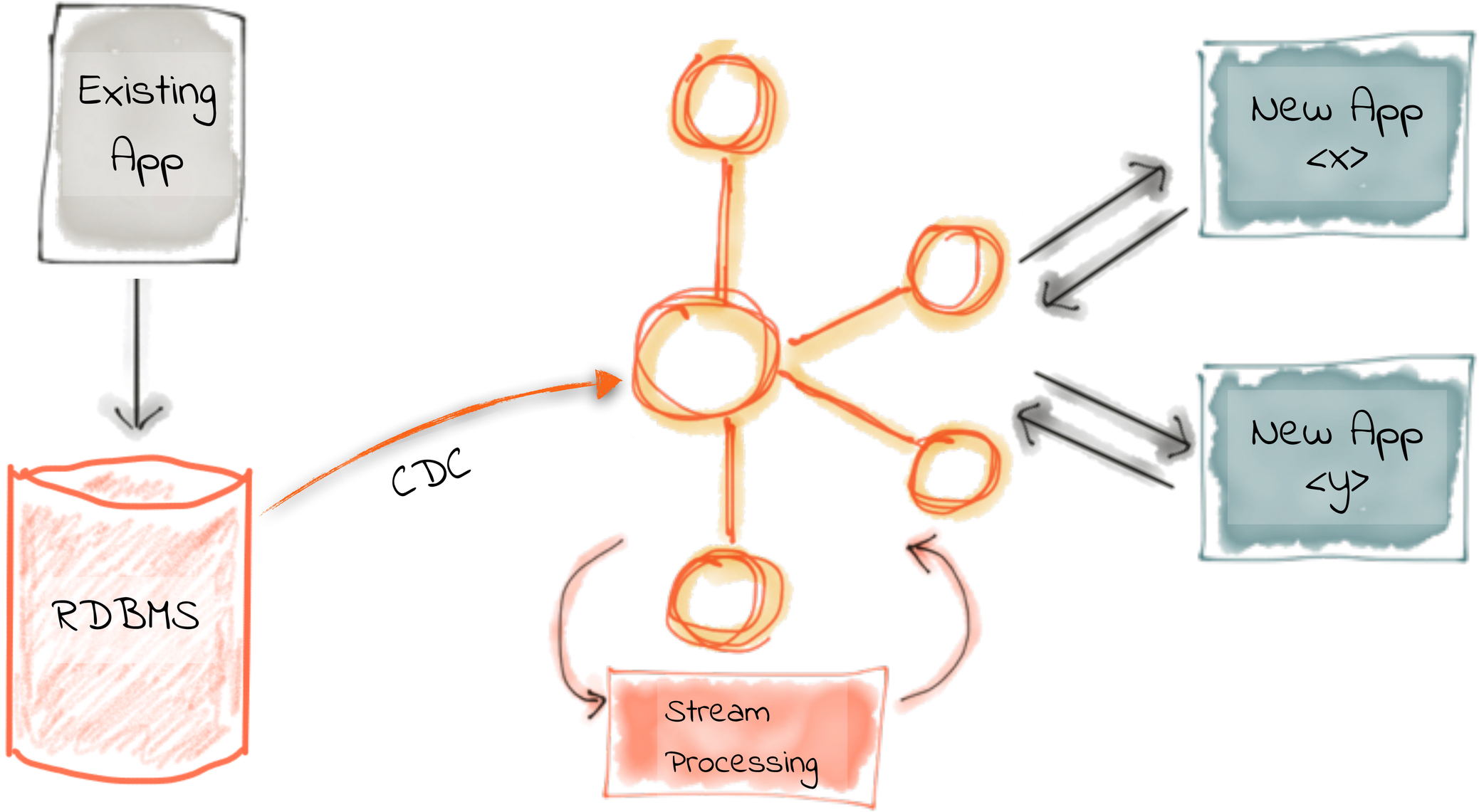
Evolve processing from old systems to new



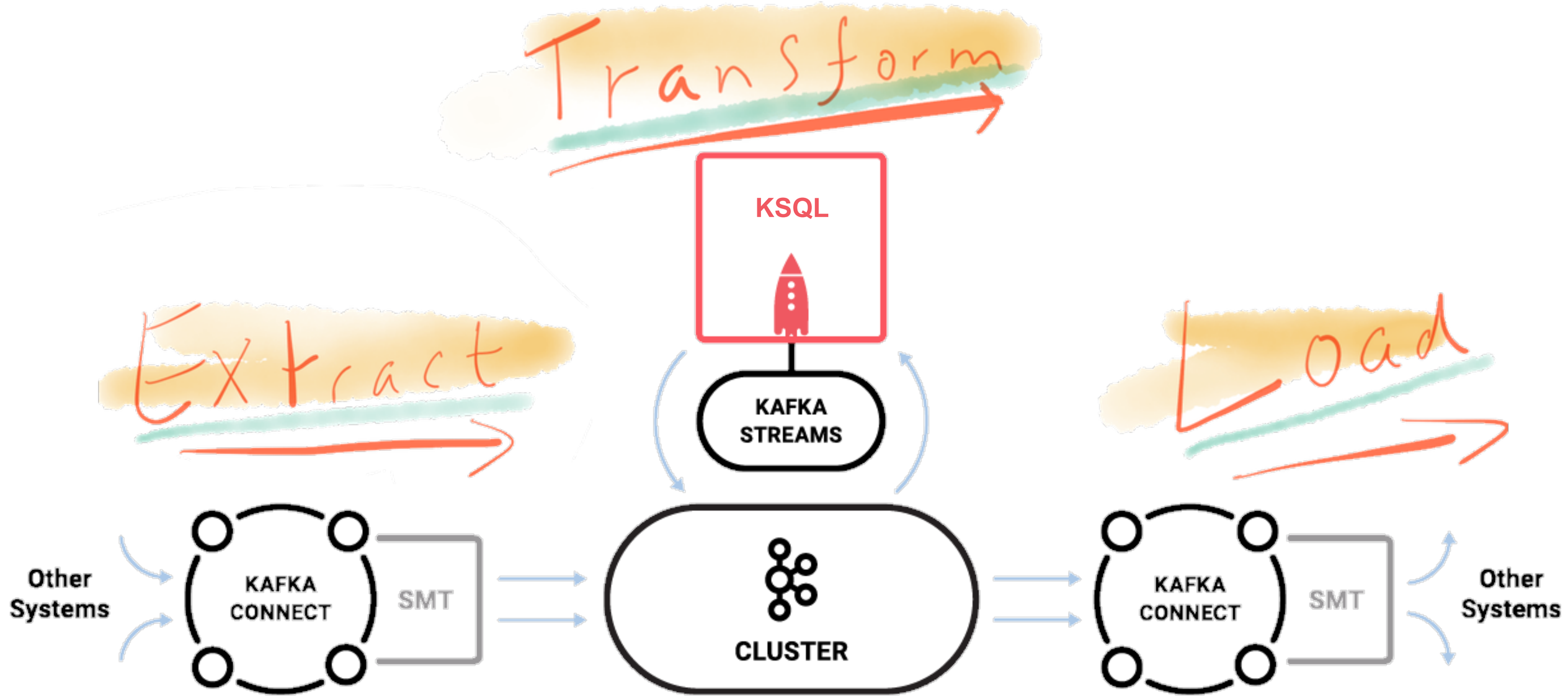
Evolve processing from old systems to new



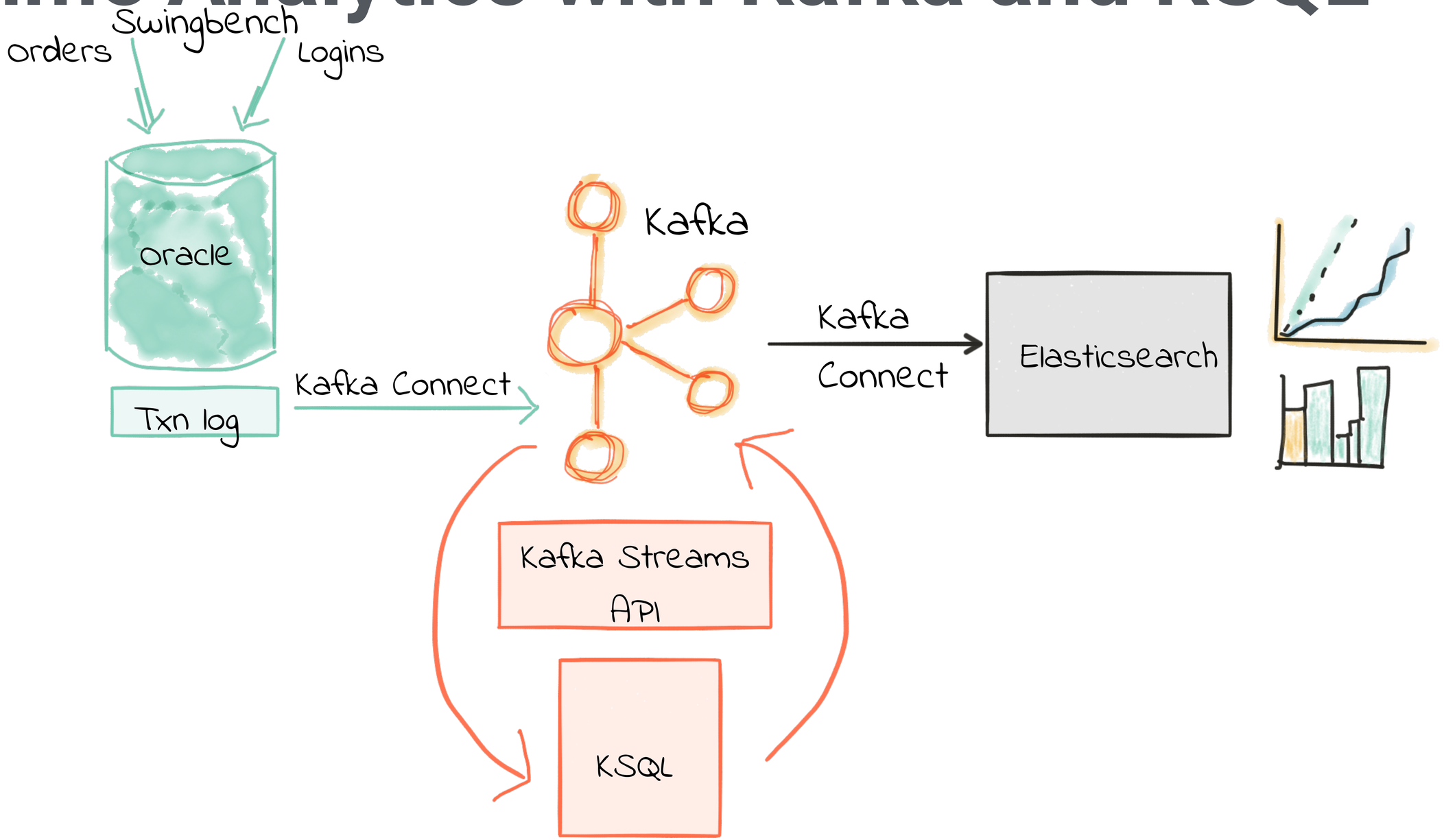
Evolve processing from old systems to new



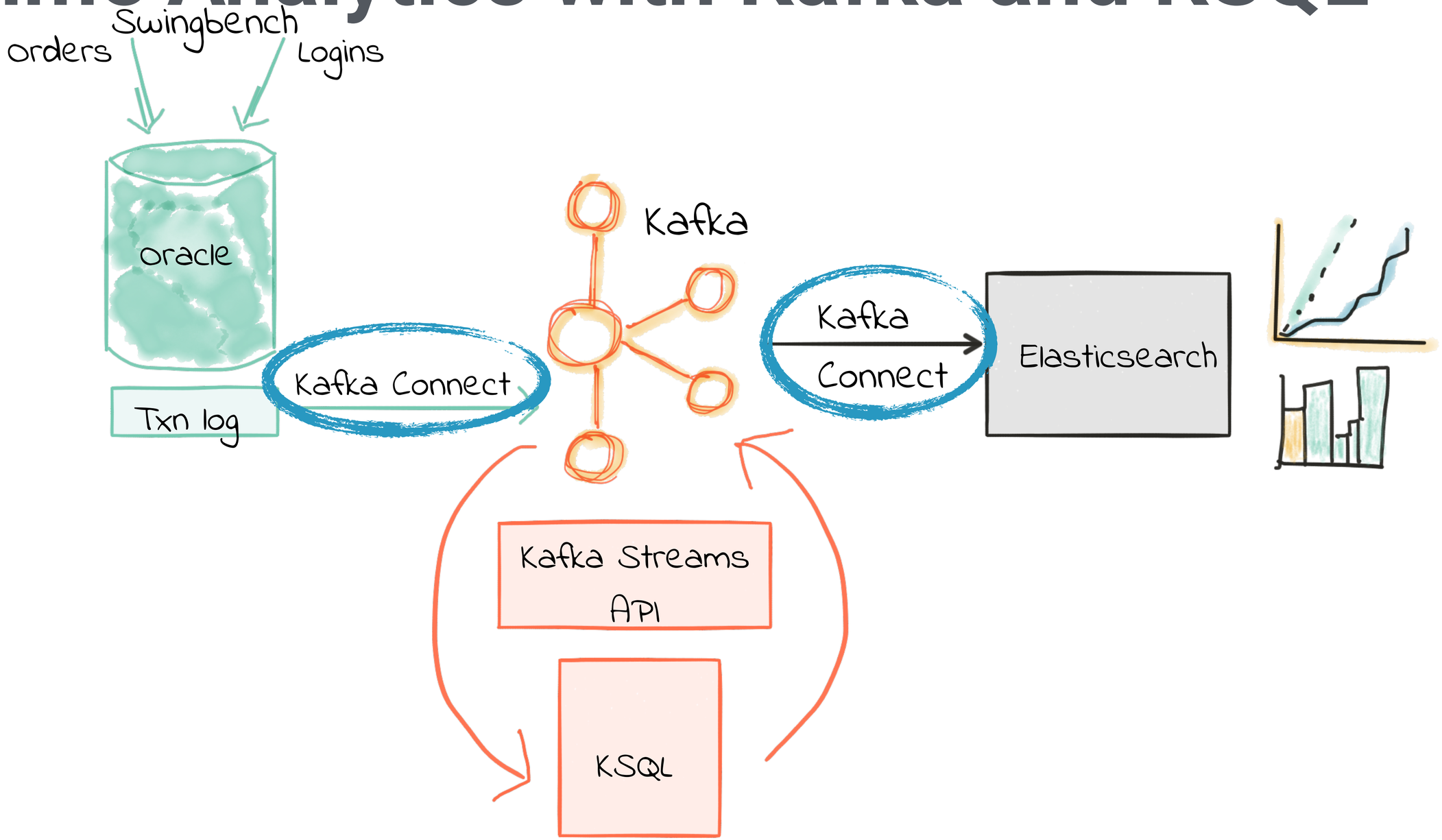
Streaming ETL, powered by Apache Kafka and Confluent Platform



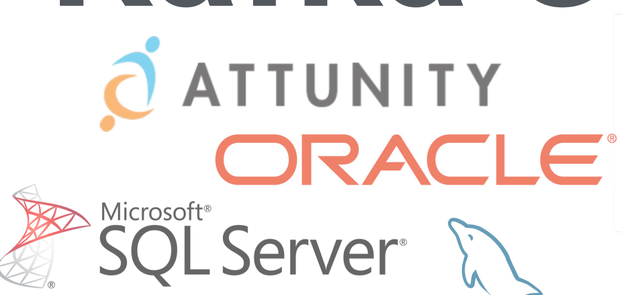
Realtime Analytics with Kafka and KSQL

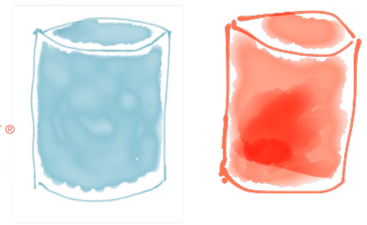


Realtime Analytics with Kafka and KSQL



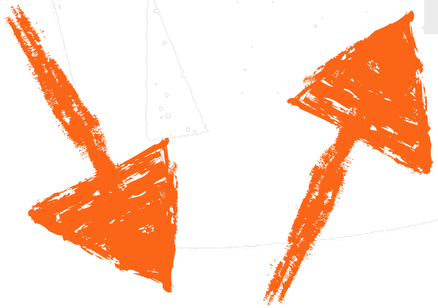
Kafka Connect

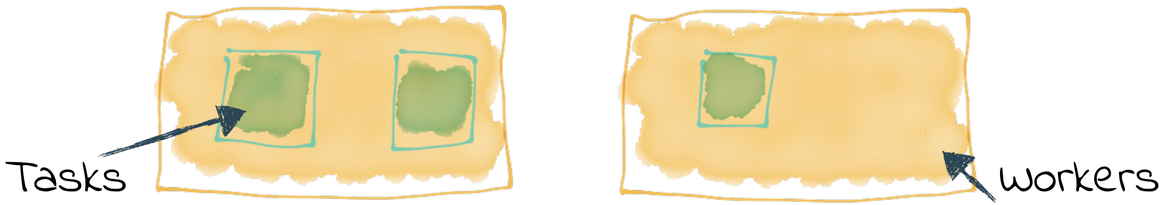




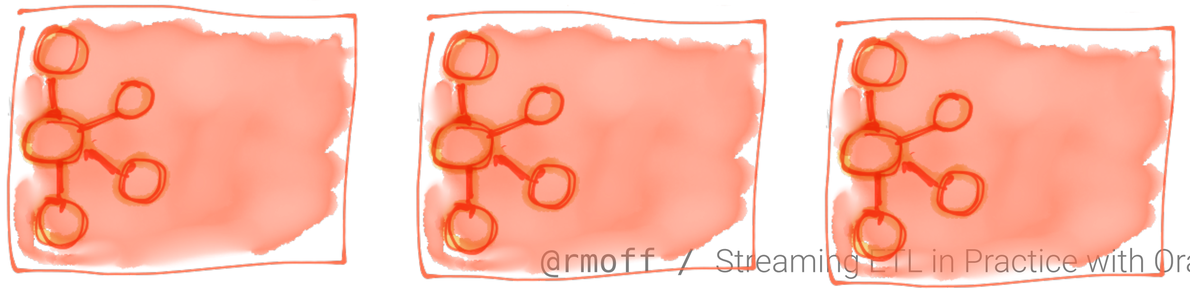
Sources

Sinks





Kafka Connect



Kafka Brokers

The Connect API of Apache Kafka®

Reliable and scalable integration of Kafka with other systems – no coding required.

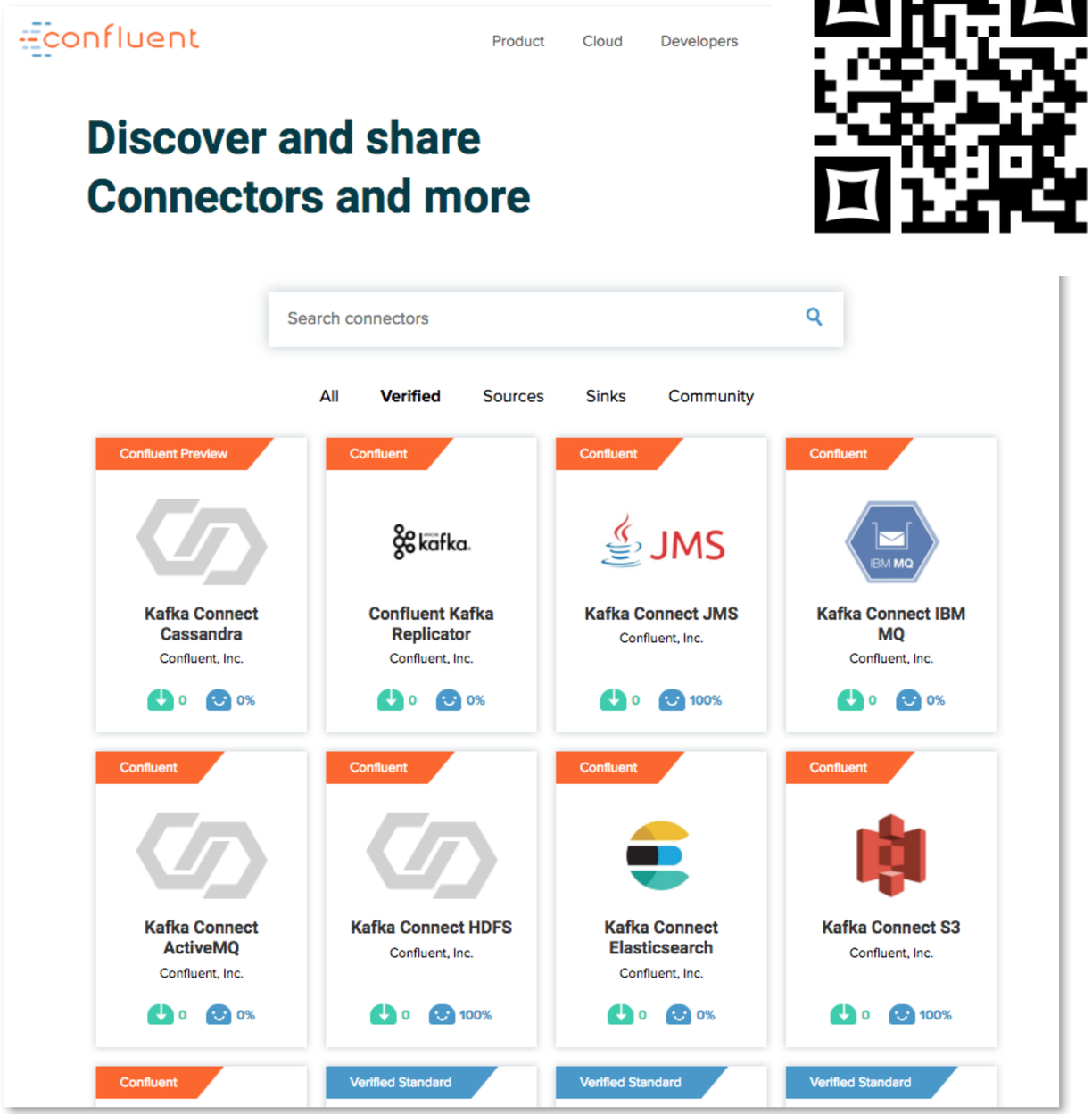
- ✓ Centralized management and configuration
- ✓ Support for hundreds of technologies including RDBMS, Elasticsearch, HDFS, S3
- ✓ Supports CDC ingest of events from RDBMS
- ✓ Preserves data schema
- ✓ Fault tolerant and automatically load balanced
- ✓ Extensible API
- ✓ Single Message Transforms
- ✓ Part of Apache Kafka, included in Confluent Open Source

```
{
"connector.class": "io.confluent.connect.jdbc.JdbcSourceConnector",
"connection.url": "jdbc:mysql://localhost:3306/demo?user=rmoff&password=foo",
"table.whitelist": "sales,orders,customers"
}
```

https://docs.confluent.io/current/connect/

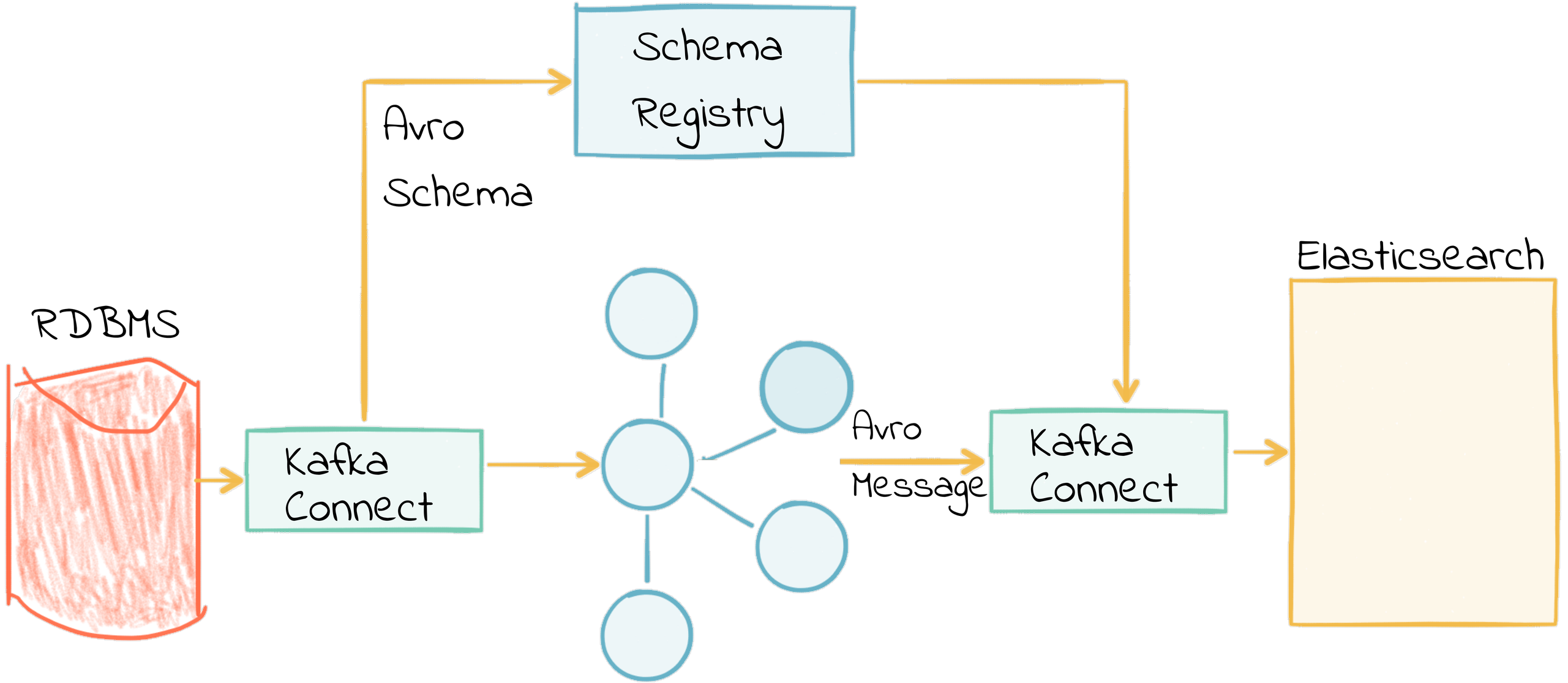
Confluent Hub

- Launched June 2018
- One-stop place to discover and download :
  - Connectors
  - Transformations
  - Converters

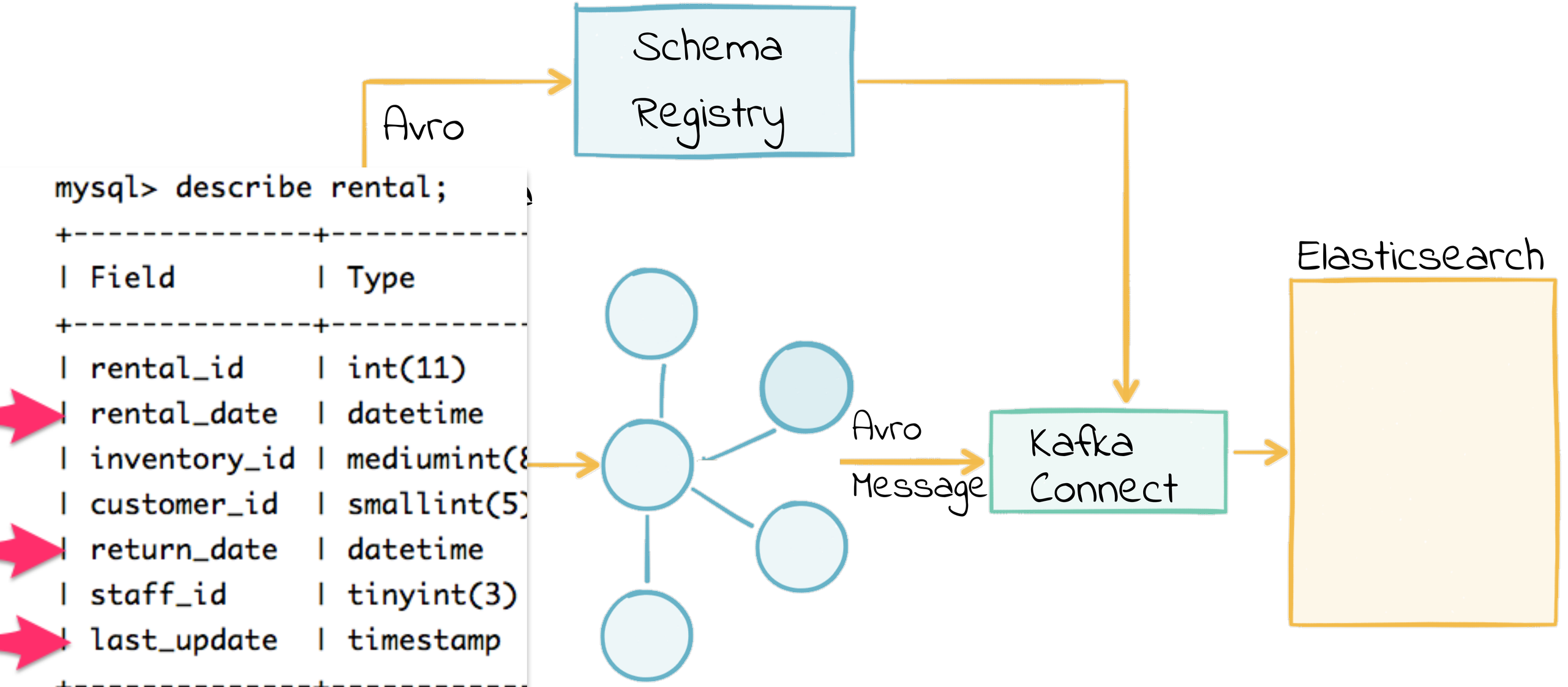


hub.confluent.io

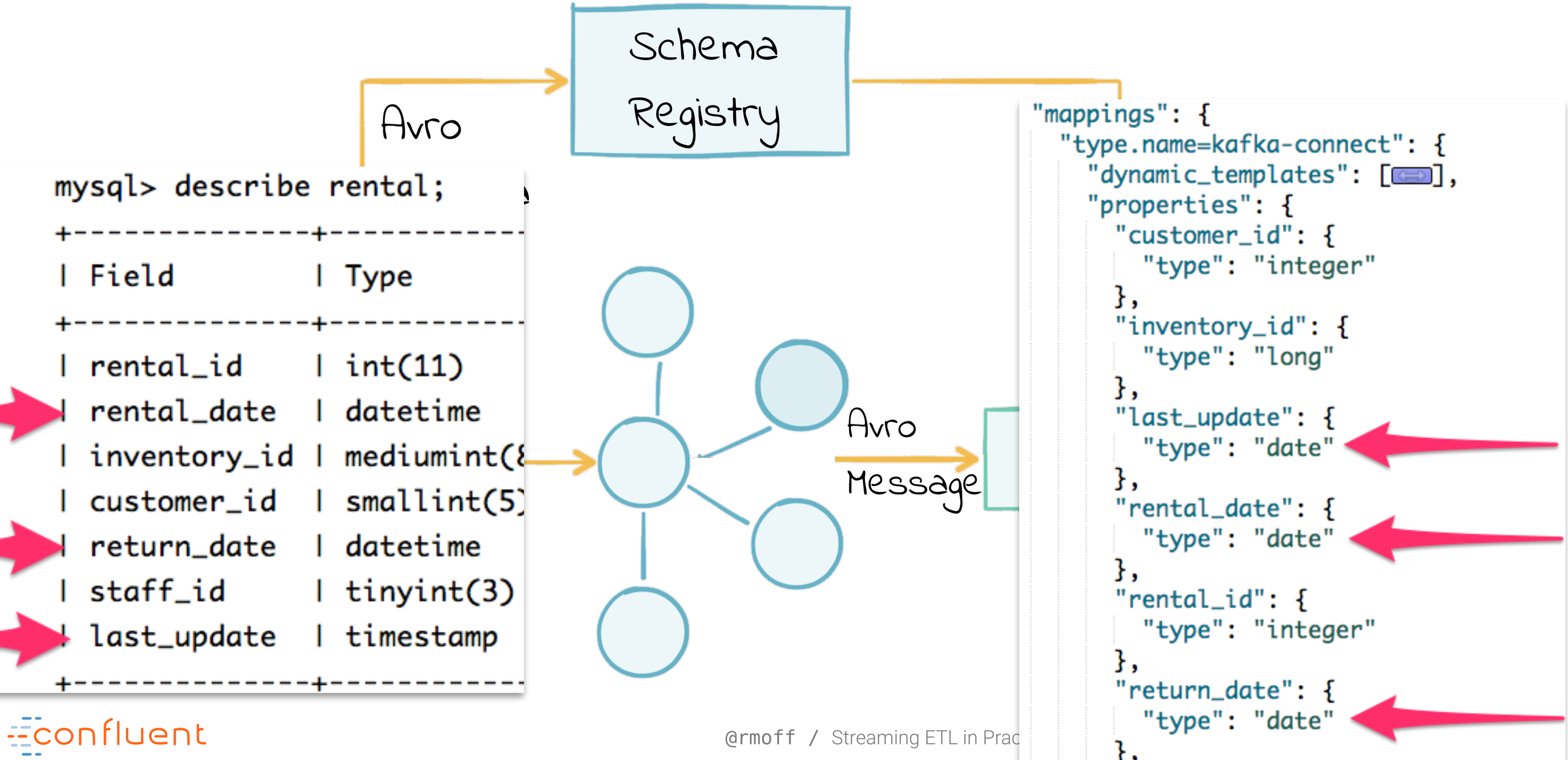
Kafka Connect + Schema Registry = WIN



Kafka Connect + Schema Registry = WIN



Kafka Connect + Schema Registry = WIN



Change-Data-Capture (CDC)

Change-Data-Capture (CDC)

- **CDC** is a generic term referring to *capturing changing data* typically from a RDBMS.
- Two general approaches:

Change-Data-Capture (CDC)

- **CDC** is a generic term referring to *capturing changing data* typically from a RDBMS.
- Two general approaches:
  - **Query-based CDC**

Change-Data-Capture (CDC)

- **CDC** is a generic term referring to *capturing changing data* typically from a RDBMS.
- Two general approaches:
  - **Query-based CDC**
  - **Log-based CDC**

There are other options including hacks with Triggers, Flashback etc but these are system and/or technology-specific.

Query-based CDC

- Use a database query to try and identify new & changed rows

```
SELECT * FROM my_table
WHERE col > <value of col last time we polled>
```

- Implemented with the open source **Kafka Connect JDBC connector**
  - Can import based on table names, schema, or bespoke SQL query
  - Incremental ingest driven through incrementing ID column and/or timestamp column

Log-based CDC

- Use the database's transaction log to identify every single change event
- Various CDC tools available that integrate with Apache Kafka (more of this later…)

```
Logdump 12 >pos 6636
Reading forward from RBA 6636
Logdump 13 >n
------------------------------------------------------------------
Hdr-Ind    :     E  (x45)     Partition  :     .  (x0c)
UndoFlag   :     .  (x00)     BeforeAfter:     A  (x41)
RecLength  :   256  (x0100)   IO Time    : 2016/09/06 11:59:23.000.589
IOType     :     5  (x05)     OrigNode   :   255  (xff)
TransInd   :     .  (x00)     FormatType :     R  (x52)
SyskeyLen  :     0  (x00)     Incomplete :     .  (x00)
AuditRBA   :         393      AuditPos   : 30266384
Continued  :     N  (x00)     RecCount   :     1  (x01)

2016/09/06 11:59:23.000.589 Insert               Len   256 RBA 6636
Name: ORCL.SOE.CUSTOMERS  (TDR Index: 3)
After  Image:                                             Partition 12   G  b
0000 000a 0000 0000 0000 0001 86a1 0001 000a 0000 | ..................
0006 616e 7477 616e 0002 000b 0000 0007 7361 6d70 | ..antwan........samp
736f 6e00 0300 0600 0000 0275 7300 0400 0b00 0000 | son........us.......
0741 4d45 5249 4341 0005 000a 0000 0000 0000 0000 | .AMERICA............
8980 0006 001d 0000 0019 616e 7477 616e 2e73 616d | ..........antwan.sam
7073 6f6e 406f 7261 636c 652e 636f 6d00 0700 0a00 | pson@oracle.com.....
0000 0000 0000 0000 9500 0800 1500 0032 3031 362d | ...............2016-
Column     0 (x0000), Len    10 (x000a)
0000 0000 0000 0001 86a1                          | ..........
Column     1 (x0001), Len    10 (x000a)
0000 0006 616e 7477 616e                          | ....antwan
Column     2 (x0002), Len    11 (x000b)
0000 0007 7361 6d70 736f 6e                       | ....sampson
Column     3 (x0003), Len     6 (x0006)
0000 0002 7573                                    | ....us
Column     4 (x0004), Len    11 (x000b)
0000 0007 414d 4552 4943 41                       | ....AMERICA
Column     5 (x0005), Len    10 (x000a)
0000 0000 0000 0000 8980                          | ..........
Column     6 (x0006), Len    29 (x001d)
0000 0019 616e 7477 616e 2e73 616d 7073 6f6e 406f | ....antwan.sampson@o
7261 636c 652e 636f 6d                            | racle.com
Column     7 (x0007), Len    10 (x000a)
0000 0000 0000 0000 0095                          | ..........
Column     8 (x0008), Len    21 (x0015)
```

Query-based vs Log-based CDC

- **Query-based**
  - +Usually easier to setup, and requires fewer permissions
  - - Needs specific columns in source schema
  - - Impact of polling the DB (or higher latencies tradeoff)
  - - Can't track deletes, or multiple events between polling interval

Read more: http://cnfl.io/kafka-cdc

Photo by Sebastian Pociecha on Unsplash

Query-based vs Log-based CDC

- **Log-based**
  - +Greater data fidelity
  - +Lower latency
  - +Lower impact on source
  - - More setup steps
  - - Higher system privileges required
  - - For propriatory databases, usually $$$

Read more: http://cnfl.io/kafka-cdc

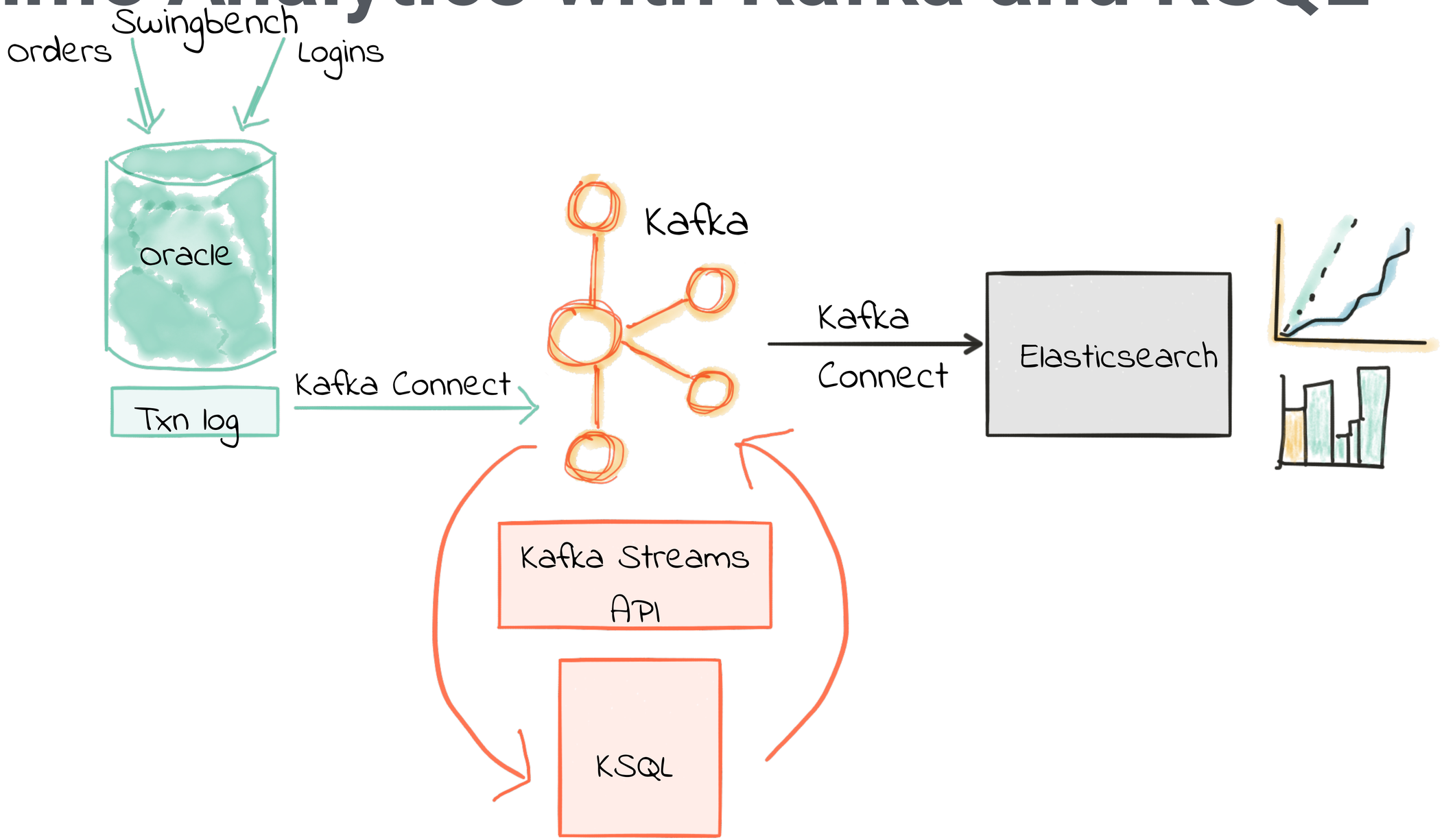
ⓘ *For query-based CDC, use the Confluent Kafka Connect JDBC connector*

Which Log-Based CDC Tool?

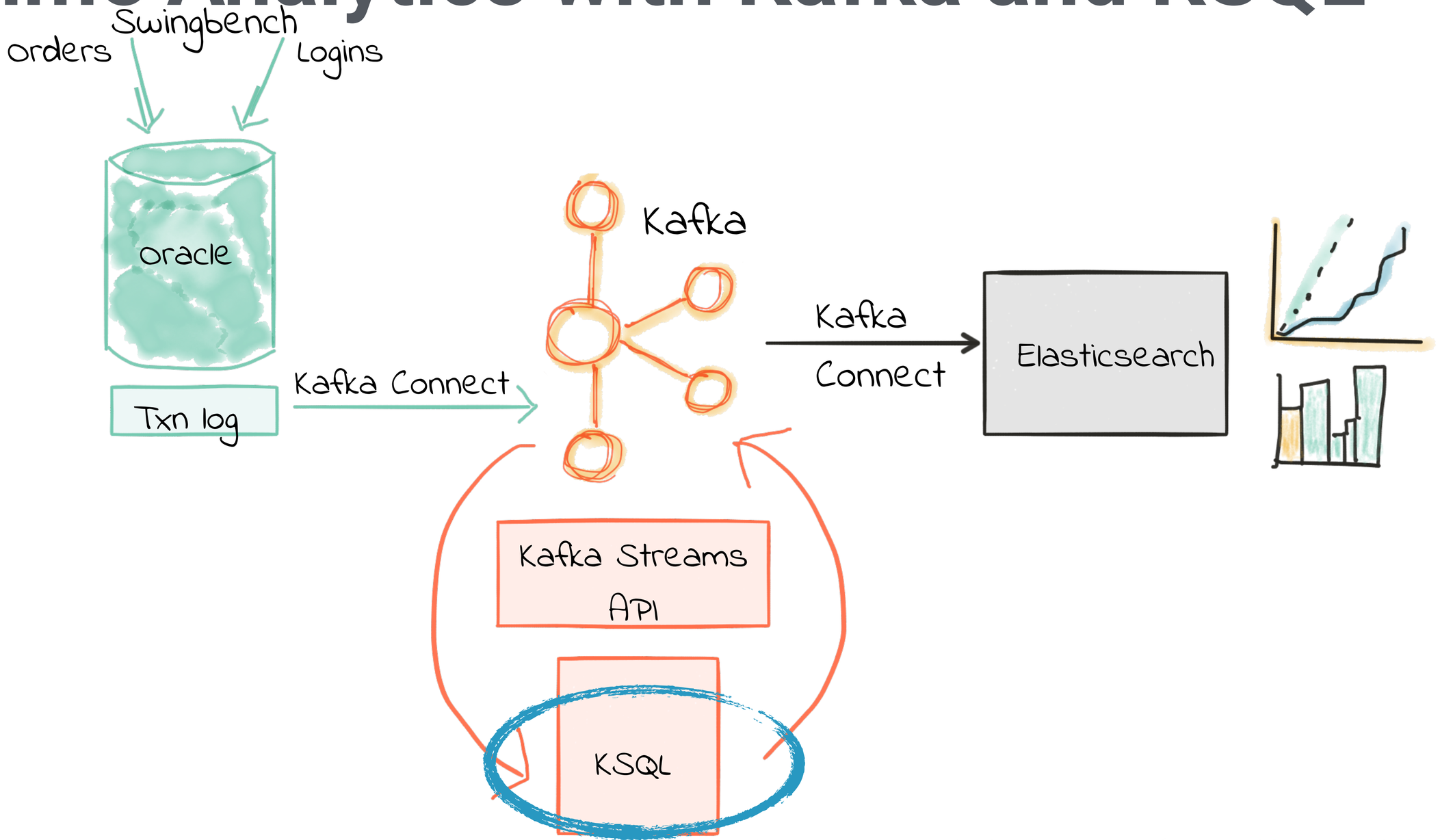
- Proprietory RDBMS, e.g. **Oracle**, **MS SQL**
  - Attunity
  - IBM InfoSphere Data Replication (IIDR)
  - Oracle GoldenGate
  - SQData
  - HVR
- Open Source RDBMS, e.g. **MySQL**, **PostgreSQL**
  - Debezium
  - (+ paid options)
- Mainframe e.g. **VSAM**, **IMS**
  - Attunity
  - SQData

All these options integrate with Apache Kafka and Confluent Platform, including support for the Schema Registry

Realtime Analytics with Kafka and KSQL



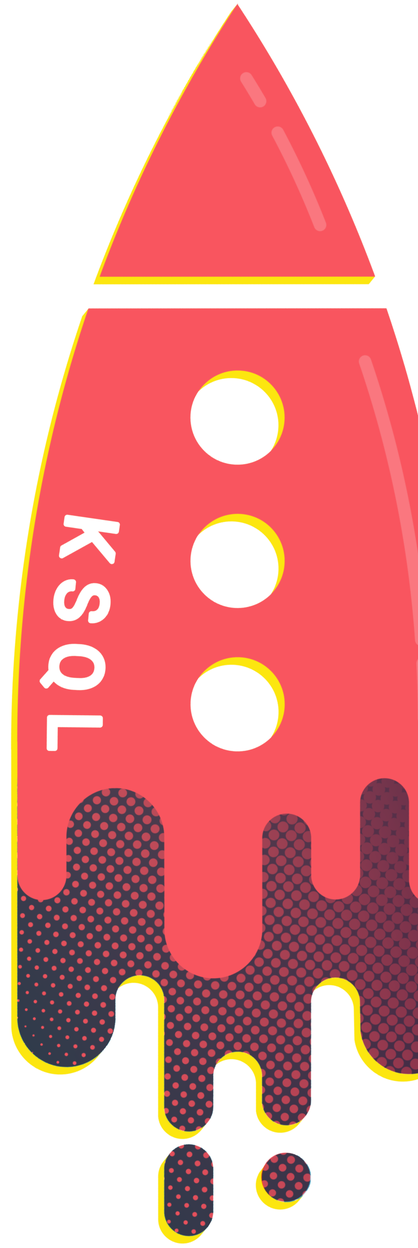
Realtime Analytics with Kafka and KSQL



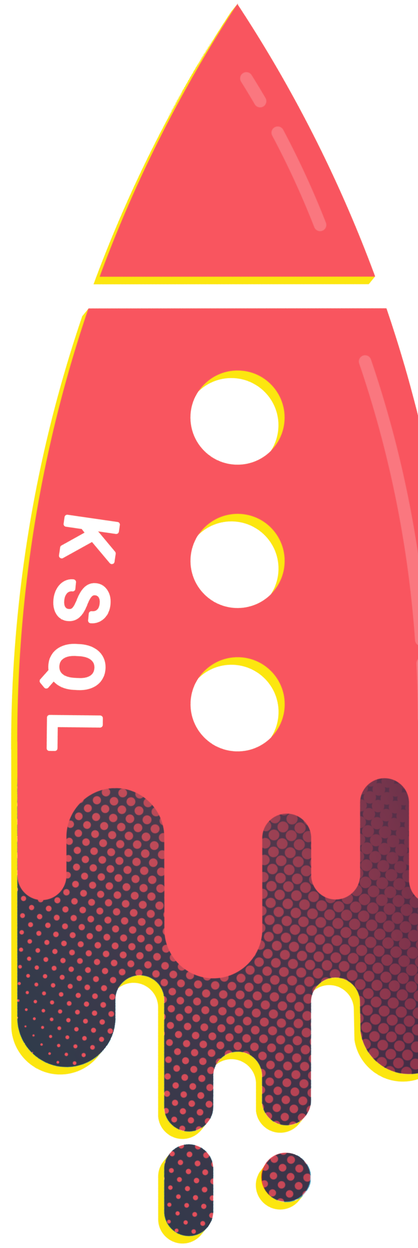
KSQL

is a

Declarative Stream Processing Language

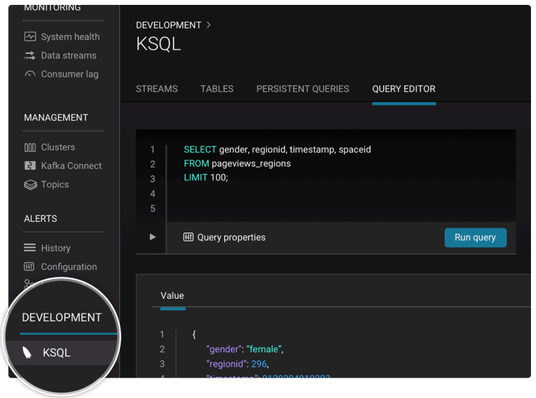


KSQL
is the
Streaming
SQL Engine
for
Apache Kafka



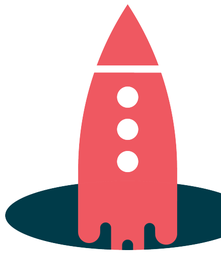
KSQL in Development and Production

Interactive KSQL
for development and testing



REST

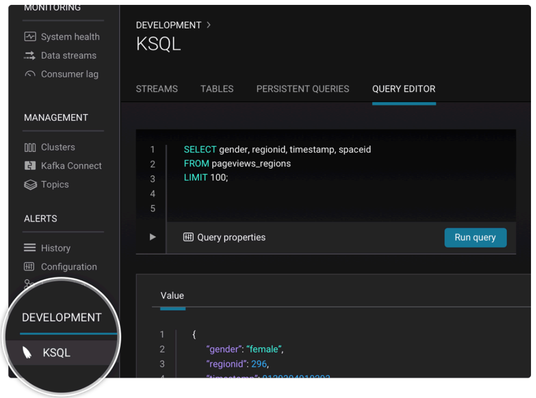
"Hmm, let me try out this idea..."



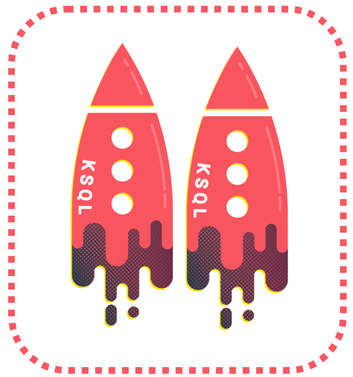
KSQL in Development and Production

Interactive KSQL
for development and testing

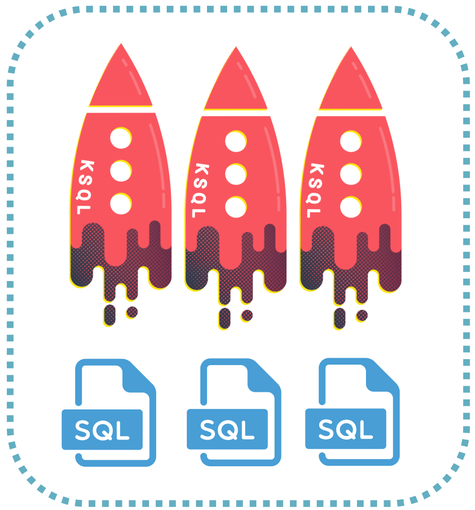
Headless KSQL
for Production



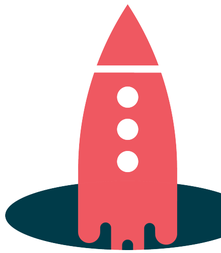
REST



Desired KSQL queries have been identified



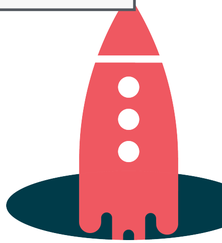
"Hmm, let me try out this idea..."



KSQL for Streaming ETL

Joining, filtering, and aggregating streams of event data

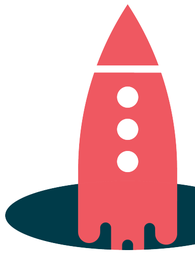
```
CREATE STREAM vip_actions AS
  SELECT userid, page, action
  FROM clickstream c
  LEFT JOIN users u
    ON c.userid = u.user_id
  WHERE u.level = 'Platinum';
```



KSQL for Anomaly Detection

Identifying patterns or anomalies in real-time data, surfaced in milliseconds

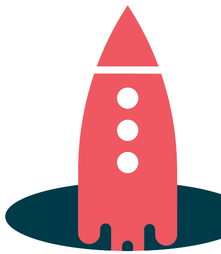
```
CREATE TABLE possible_fraud AS
  SELECT card_number, count(*)
    FROM authorization_attempts
    WINDOW TUMBLING (SIZE 5 SECONDS)
    GROUP BY card_number
    HAVING count(*) > 3;
```



KSQL for Real-Time Monitoring

- Log data monitoring, tracking and alerting
- syslog data
- Sensor / IoT data

```
CREATE STREAM SYSLOG_INVALID_USERS AS
SELECT HOST, MESSAGE
FROM SYSLOG
WHERE MESSAGE LIKE '%Invalid user%';
```



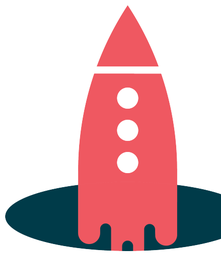
http://cnfl.io/syslogs-filtering / http://cnfl.io/syslog-alerting

confluent

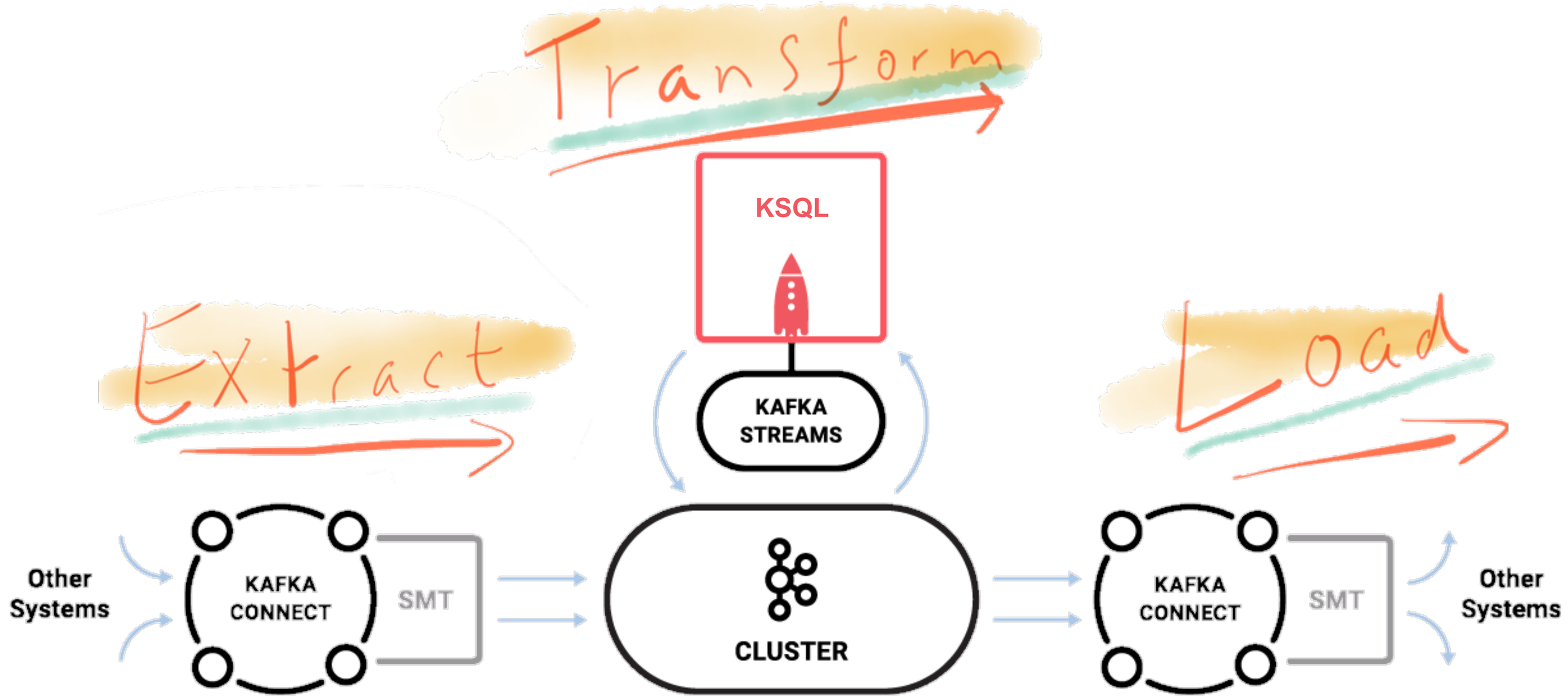
KSQL for Data Transformation

Make simple derivations of existing topics from the command line

```
CREATE STREAM views_by_userid
  WITH (PARTITIONS=6, REPLICAS=5,
        VALUE_FORMAT='AVRO',
        TIMESTAMP='view_time') AS
  SELECT * FROM clickstream
  PARTITION BY user_id;
```

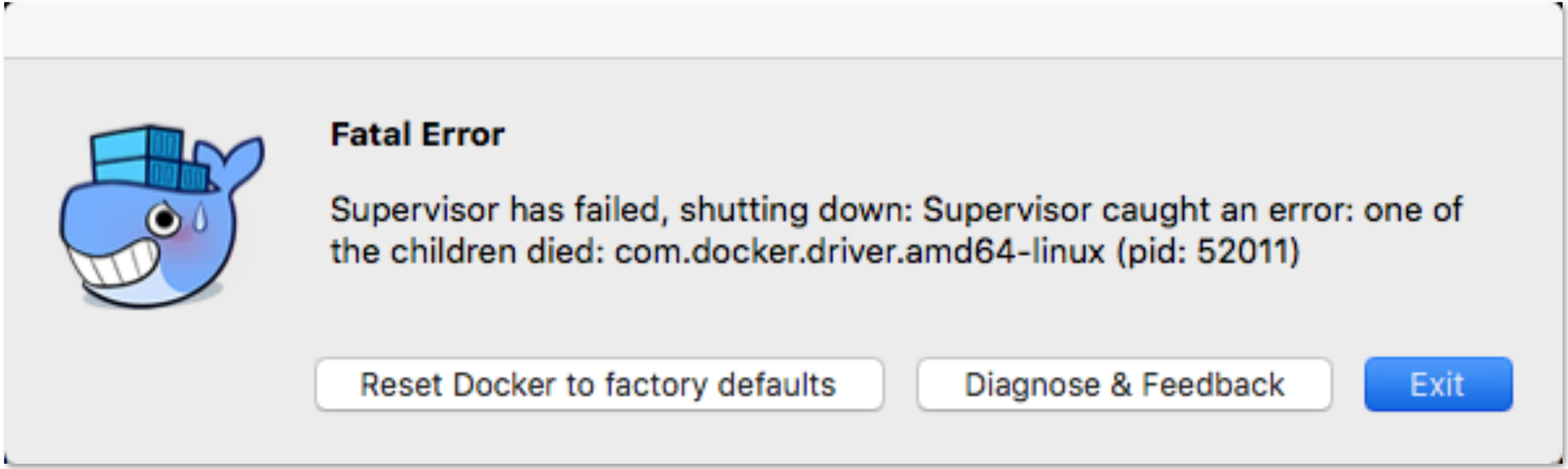


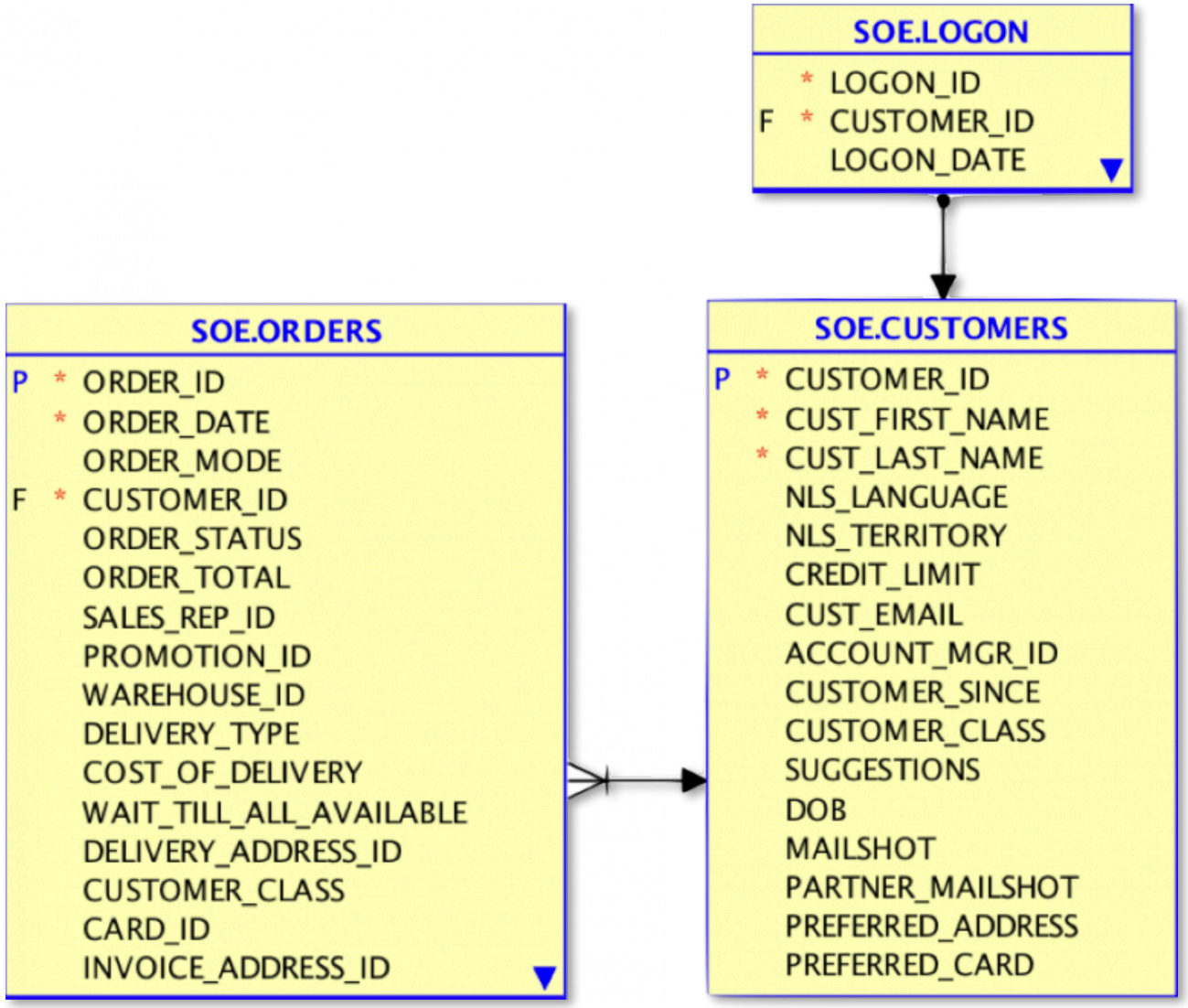
Streaming ETL, powered by Apache Kafka and Confluent Platform

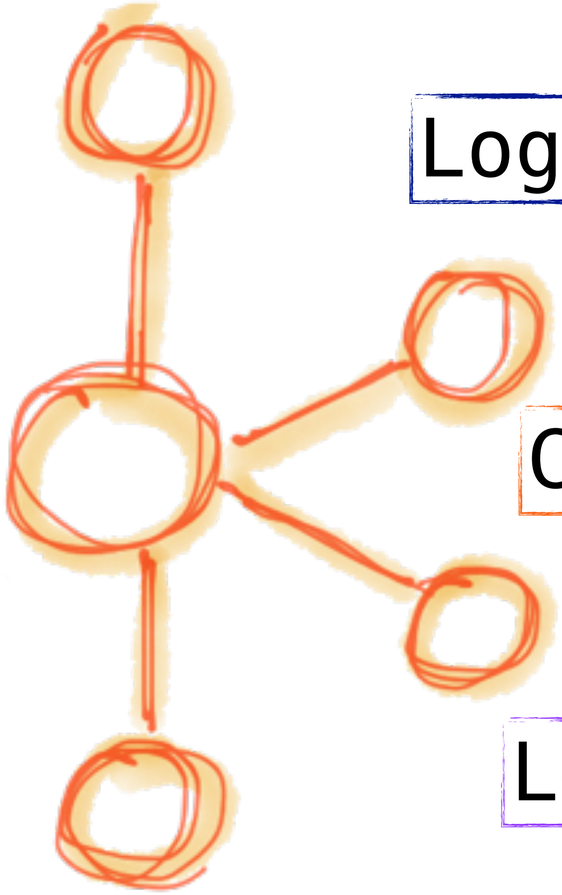


Demo Time!

Demo Time!


Fatal Error
Supervisor has failed, shutting down: Supervisor caught an error: one of the children died: com.docker.driver.amd64-linux (pid: 52011)
Reset Docker to factory defaults
Diagnose & Feedback
Exit


SOE.LOGON
* LOGON_ID
F * CUSTOMER_ID
LOGON_DATE
SOE.ORDERS
P * ORDER_ID
* ORDER_DATE
ORDER_MODE
F * CUSTOMER_ID
ORDER_STATUS
ORDER_TOTAL
SALES_REP_ID
PROMOTION_ID
WAREHOUSE_ID
DELIVERY_TYPE
COST_OF_DELIVERY
WAIT_TILL_ALL_AVAILABLE
DELIVERY_ADDRESS_ID
CUSTOMER_CLASS
CARD_ID
INVOICE_ADDRESS_ID
SOE.CUSTOMERS
P * CUSTOMER_ID
* CUST_FIRST_NAME
* CUST_LAST_NAME
NLS_LANGUAGE
NLS_TERRITORY
CREDIT_LIMIT
CUST_EMAIL
ACCOUNT_MGR_ID
CUSTOMER_SINCE
CUSTOMER_CLASS
SUGGESTIONS
DOB
MAILSHOT
PARTNER_MAILSHOT
PREFERRED_ADDRESS
PREFERRED_CARD

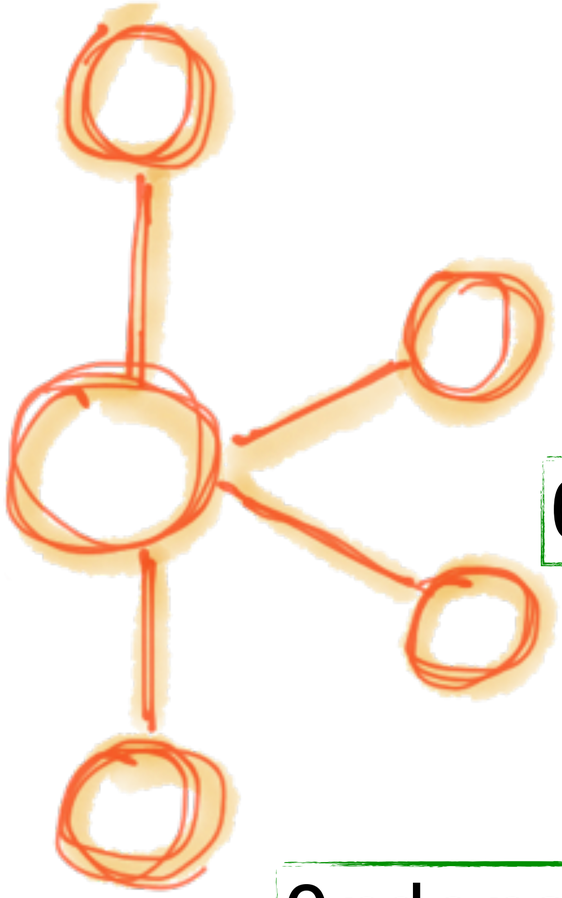


Logons

Customers

Logon_Enriched

Join, and Derive additional columns



Orders

Aggregate order metrics by hour

Orders_agg_hourly

```
780549 | 504856 | monty blackburn | 1431788679000 | Ocasional | 2591.0
780550 | 504857 | garry rhodes | 1463345631000 | Ocasional | 3339.0
780587 | 504859 | ivan stevens | 1494902584000 | Ocasional | 3205.0
780620 | 504861 | rupert summers | 1494902585000 | Ocasional | 4055.0
780655 | 504866 | darnell goodwin | 1463345635000 | Ocasional | 2721.0
780650 | 504864 | phillip vazquez | 1431788682000 | Ocasional | 2643.0
780653 | 504865 | mike atkinson | 1494902586000 | Ocasional | 2270.0
780690 | 504876 | taylor stewart | 1431788684000 | Ocasional | 4673.0
780685 | 504874 | jonathon cline | 1463345635000 | Ocasional | 4235.0
780687 | 504875 | carter patel | 1463345635000 | Ocasional | 496.0
^CQuery terminated
ksql> CREATE STREAM LOGON_ENRICHED AS SELECT L.LOGON_ID, C.CUSTOMER_ID, C.CUST_FIRST_NAME + ' ' + C.CUST_LAST_NAME AS
 CUST_FULL_NAME, \
>        C.CUSTOMER_SINCE, C.CUSTOMER_CLASS, C.CREDIT_LIMIT \
> FROM LOGON L \
>      LEFT OUTER JOIN CUSTOMERS C \
>      ON L.CUSTOMER_ID = C.CUSTOMER_ID \
> WHERE C.CUSTOMER_ID IS NOT NULL;
```

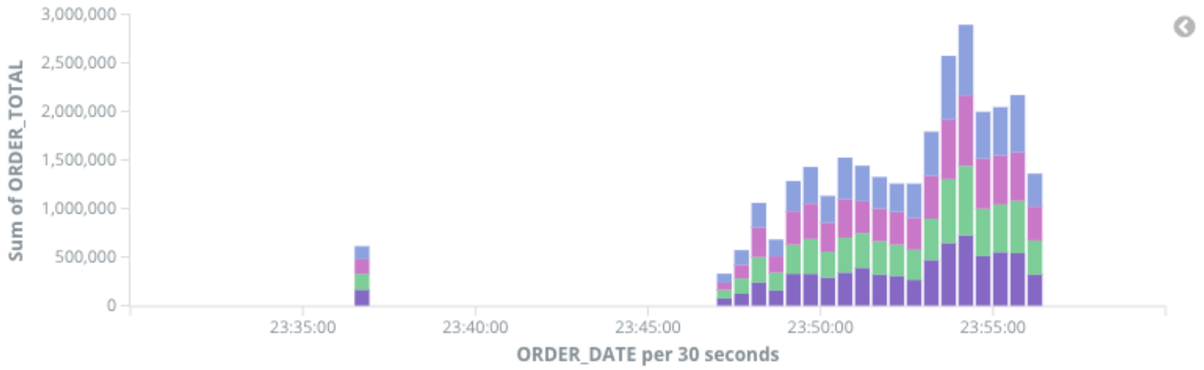
```
780549 | 504856 | monty blackburn | 1431788679000 | Ocasional | 2591.0
780550 | 504857 | garry rhodes | 1463345631000 | Ocasional | 3339.0
780587 | 504859 | ivan stevens | 1494902584000 | Ocasional | 3205.0
780620 | 504861 | rupert summers | 1494902585000 | Ocasional | 4055.0
780655 | 504866 | darnell goodwin | 1463345635000 | Ocasional | 2721.0
780650 | 504864 | phillip vazquez | 1431788682000 | Ocasional | 2643.0
780653 | 504865 | mike atkinson | 1494902586000 | Ocasional | 2270.0
780690 | 504876 | taylor stewart | 1431788684000 | Ocasional | 4673.0
780685 | 504874 | jonathon cline | 1463345635000 | Ocasional | 4235.0
780687 | 504875 | carter patel | 1463345635000 | Ocasional | 496.0
^CQuery terminated
ksql> CREATE STREAM LOGON_ENRICHED AS SELECT L.LOGON_ID, C.CUSTOMER_ID, C.CUST_FIRST_NAME + ' ' + C.CUST_LAST_NAME AS
 CUST_FULL_NAME, \
>        C.CUSTOMER_SINCE, C.CUSTOMER_CLASS, C.CREDIT_LIMIT \
> FROM LOGON L \
>      LEFT OUTER JOIN CUSTOMERS C \
>      ON L.CUSTOMER_ID = C.CUSTOMER_ID \
> WHERE C.CUSTOMER_ID IS NOT NULL;
```

Title

Realtime Analytics with Oracle, Apache Kafka, KSQL, and Elasticsearch

@rmoff / 15 May 2018

Order Value per Status



Orders

1–50 of 27,996

Time	ORDER_ID	ORDER_MODE	ORDER_TOTAL
May 15th 2018, 23:56:20.000	170,918	online	1,572
May 15th 2018, 23:56:20.000	170,919	online	312
May 15th 2018, 23:56:20.000	170,920	online	0
May 15th 2018, 23:56:20.000	170,921	online	534
May 15th 2018, 23:56:20.000	170,922	online	2,702

Logons

1–50 of 321

Time	CUST_FULL_NAME	CUSTOMER_CLASS	CREDIT_LIMIT
May 15th 2018, 23:33:44.010	armand arias	Ocasional	1,874
May 15th 2018, 23:33:44.010	phillip morgan	Ocasional	2,403
May 15th 2018, 23:33:44.010	milo brewer	Ocasional	3,379
May 15th 2018, 23:33:44.010	myles rosales	Ocasional	188
May 15th 2018, 23:33:42.990	miguel gray	Ocasional	3,926

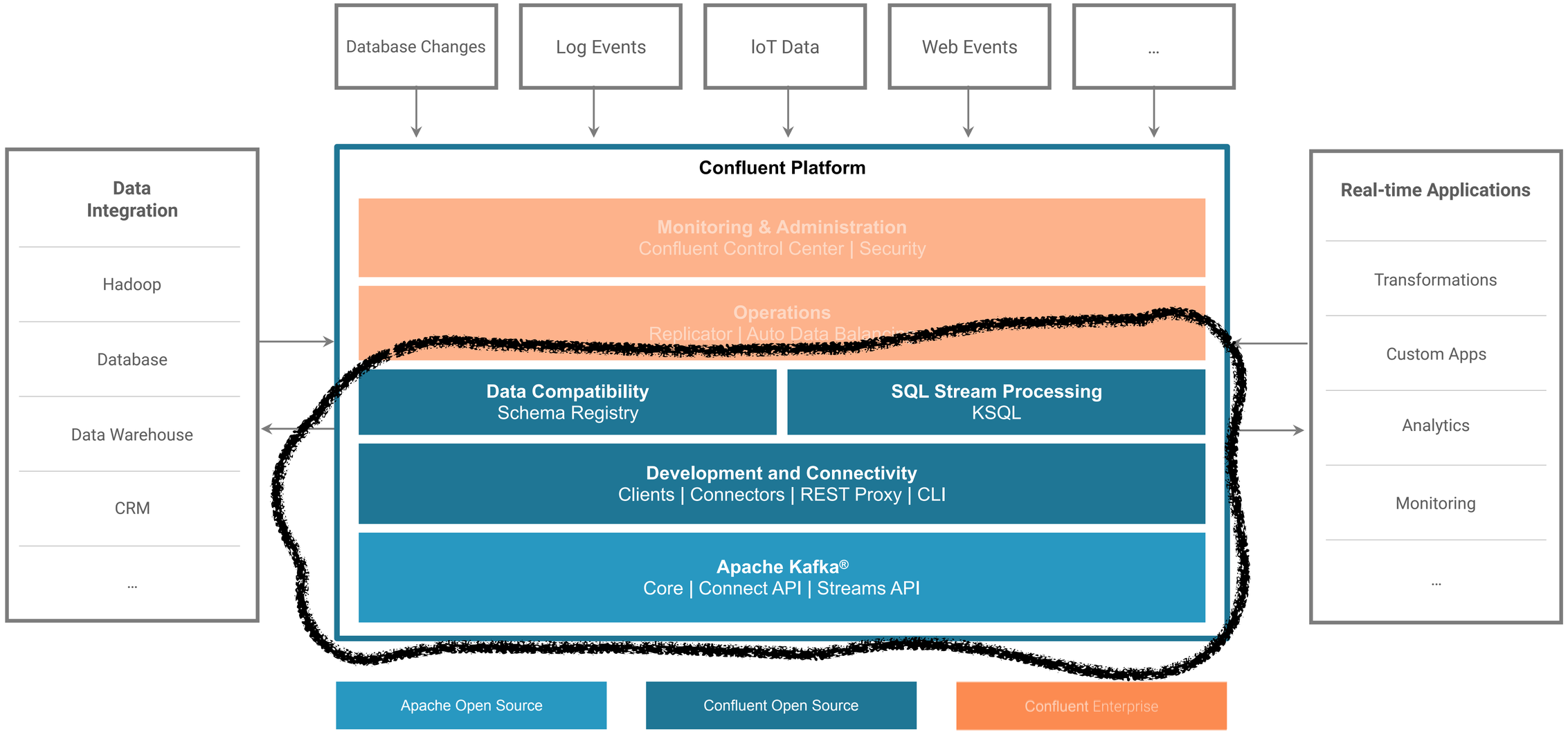
Order Stats

27,996
Count

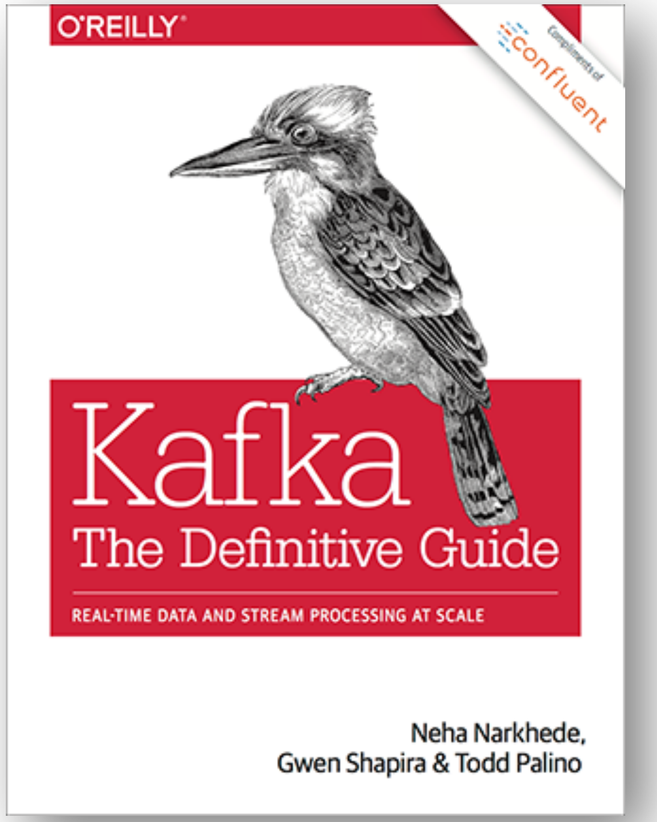
28,882,607
Sum of ORDER_TOTAL

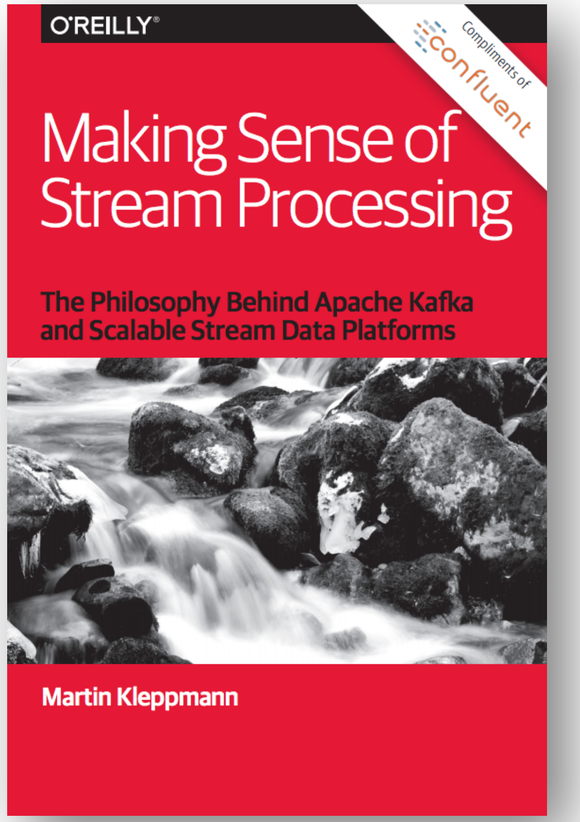
1,055.778
50th percentile of ORDER_TOTAL

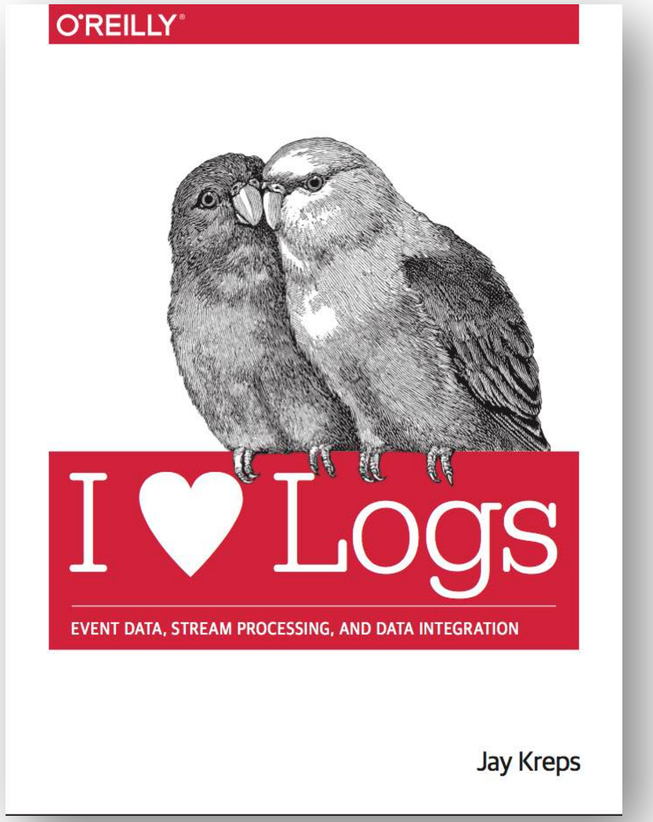
Confluent Open Source : Apache Kafka with a bunch of cool stuff! For free!



Free Books!







https://www.confluent.io/apache-kafka-stream-processing-book-bundle

https://www.confluent.io/download/
https://slackpass.io/confluentcommunity
@rmoff robin@confluent.io

#EOF